# 6track4collimation: recent updates and old features

Valentina Previtali

let me introduce you to

# Sixtrack

different people, different codes, different purposes
- and sometimes little communication

the risk is to end up with something like this...

let me introduce you to

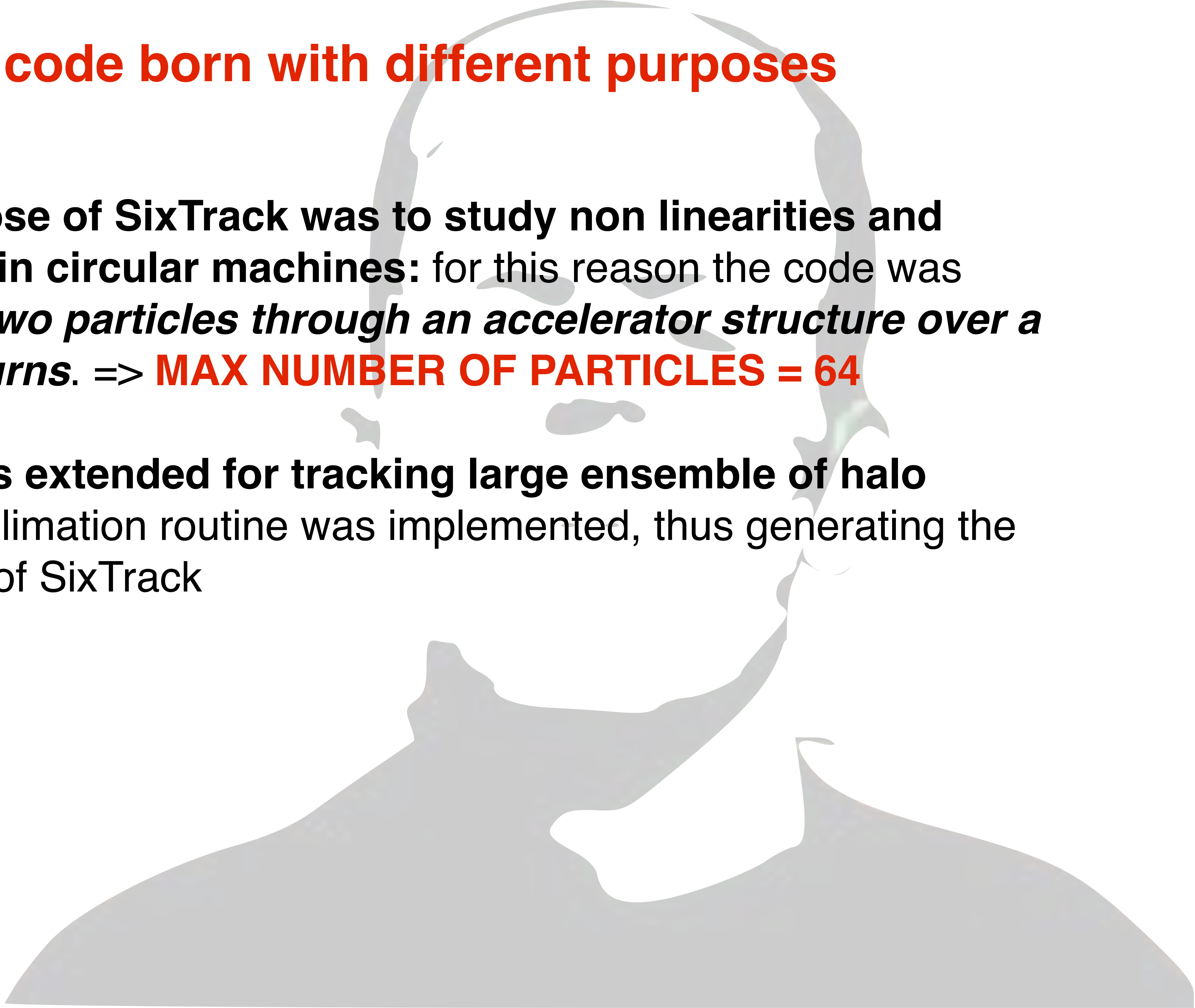
# Sixtrack

different people, different codes, different purposes - and sometimes little communication

the risk is to end up with something like this...

*huge ugly collection of not uniform pieces, impossible to maintain.*

# A code born with different purposes



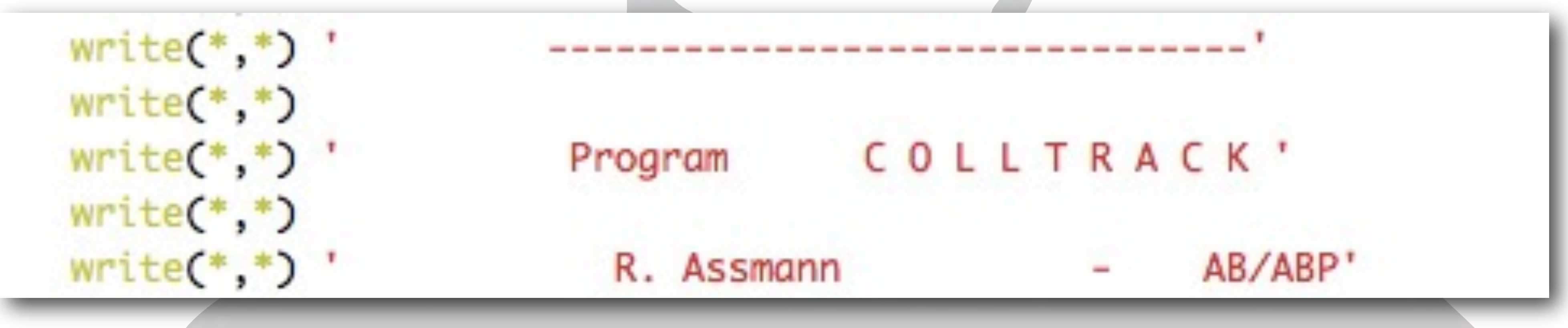
**The original purpose of SixTrack was to study non linearities and dynamic aperture in circular machines:** for this reason the code was optimized to carry ***two particles through an accelerator structure over a large number of turns***. => MAX NUMBER OF PARTICLES = 64

**Later the code was extended for tracking large ensemble of halo particles**, and a collimation routine was implemented, thus generating the collimation version of SixTrack

# A code born with different purposes

**The original purpose of SixTrack was to study non linearities and dynamic aperture in circular machines:** for this reason the code was optimized to carry ***two particles through an accelerator structure over a large number of turns***. => MAX NUMBER OF PARTICLES = 64

**Later the code was extended for tracking large ensemble of halo particles**, and a collimation routine was implemented, thus generating the collimation version of SixTrack



```
      write(*,*) '         -------------------------------'
      write(*,*)
      write(*,*) '          Program      C O L L T R A C K '
      write(*,*)
      write(*,*) '            R. Assmann           -      AB/ABP'
```

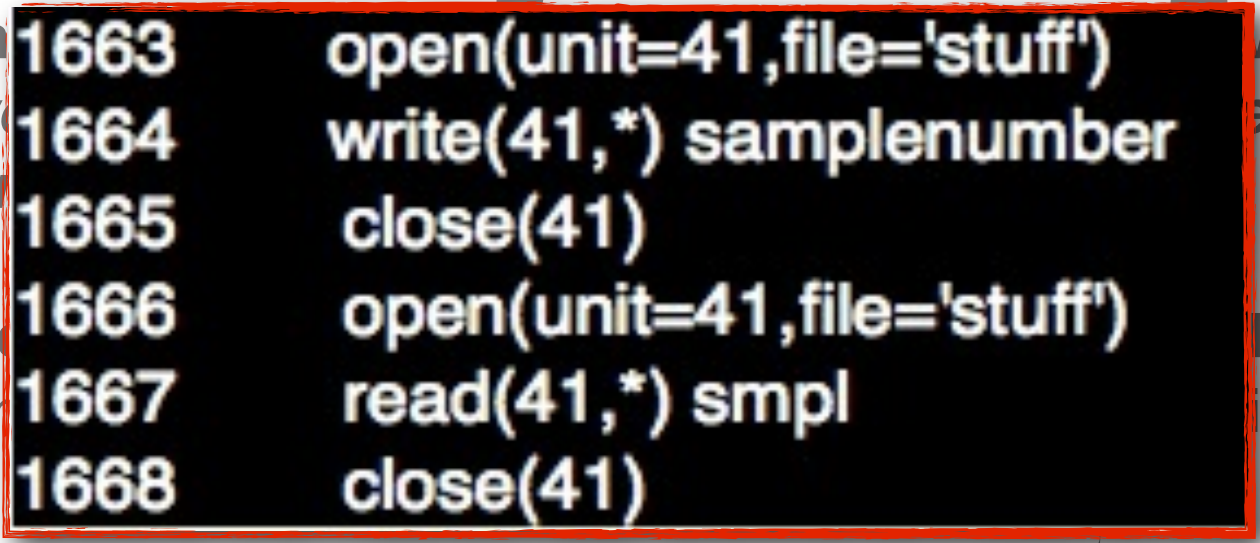
**The Monte Carlo scattering routines in the collimation code are based on the older K2 code**

ugly?

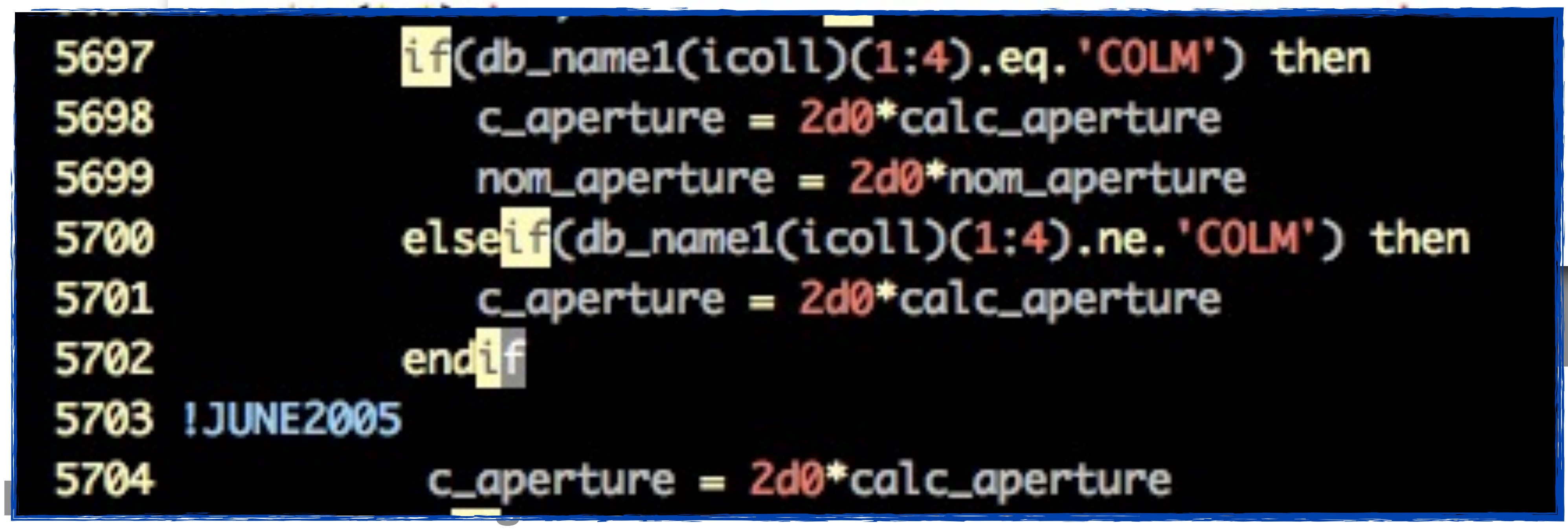
# A code born with different purposes

**The original purpose of SixTrack was to study non linearities and dynamic aperture in** was optimized to carry ***tw*** ***e over a large number of tur***

**Later the code was** **lo particles**, and a colli ing the collimation version of



```
open(unit=41,file='stuff')
write(41,*) samplenumber
 close(41)
 open(unit=41,file='stuff')
 read(41,*) smpl
 close(41)
```



```
          if(db_name1(icoll)(1:4).eq.'COLM') then
             c_aperture = 2d0*calc_aperture
             nom_aperture = 2d0*nom_aperture
          elseif(db_name1(icoll)(1:4).ne.'COLM') then
             c_aperture = 2d0*calc_aperture
          endif
!JUNE2005
           c_aperture = 2d0*calc_aperture
```

**The**
**based on the older K2 code**

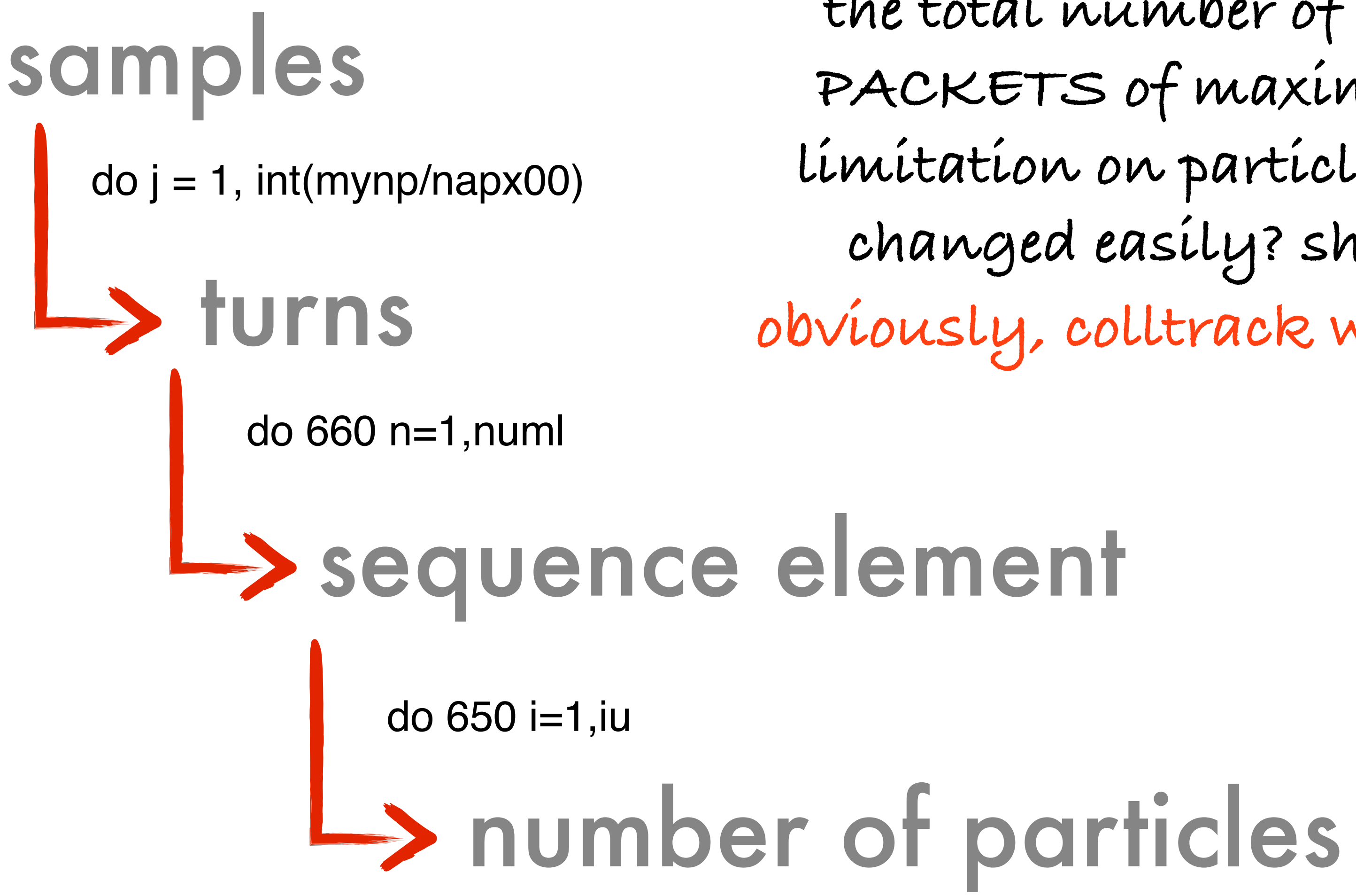
# Goal of the presentation

- Give a short overview of the collimation routine in Sixtrack
- Present the main routine included in 6track after the last official release
- present the minor changes made in the code

# general 6track4collimation structure

(starting from the TRAUTHIN routine)

# (very) general structure



samples

do j = 1, int(mynp/napx00)

turns

do 660 n=1,numl

sequence element

do 650 i=1,iu

number of particles

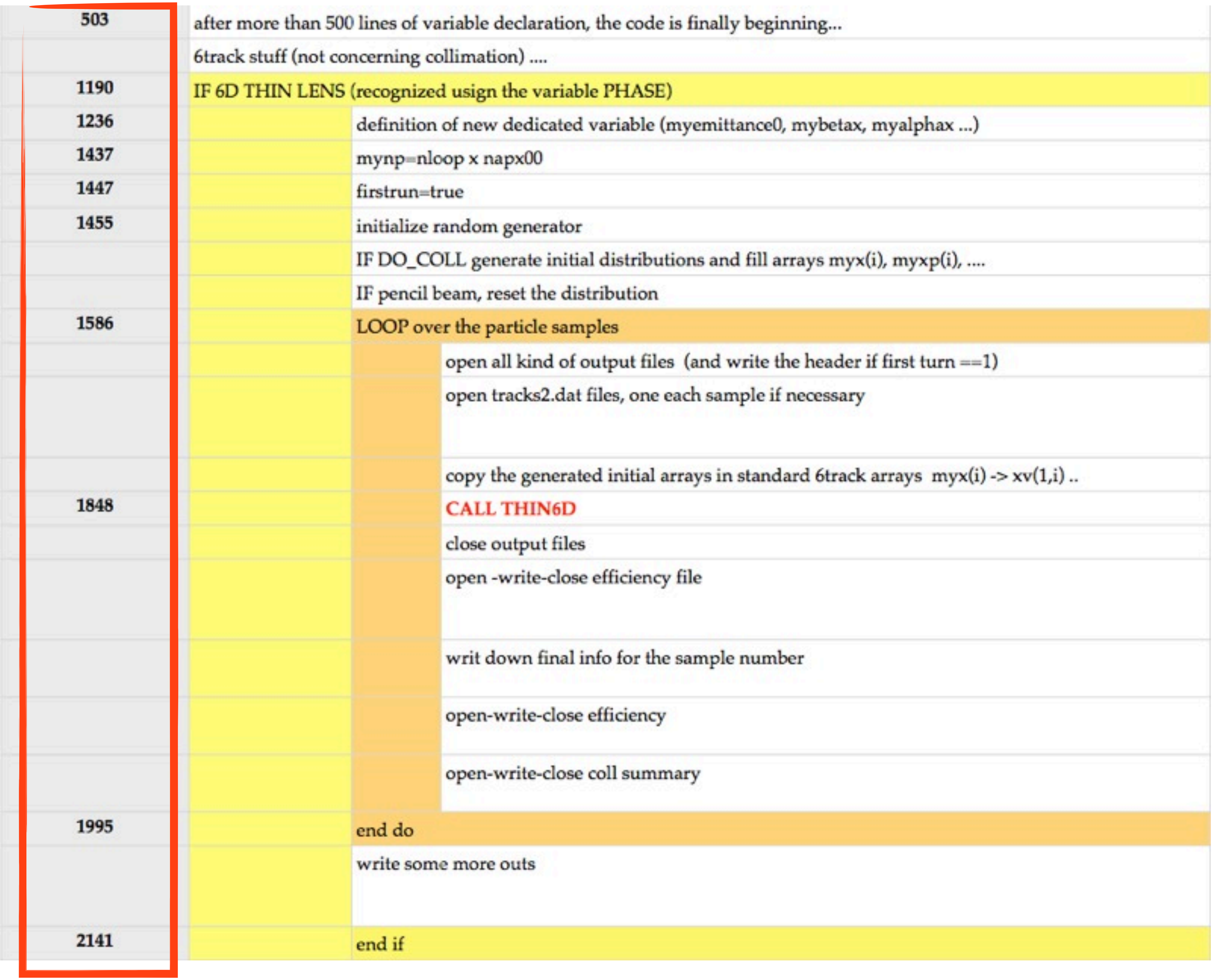
the total number of particles is divided in PACKETS of maximum 64 particles. Old limitation on particle numbers. Can this be changed easily? should be investigated.

obviously, colltrack was not structured for this...

for collimation, done within the collimate2 routine

# track.f –> thauthin

line # - as in the last DB version

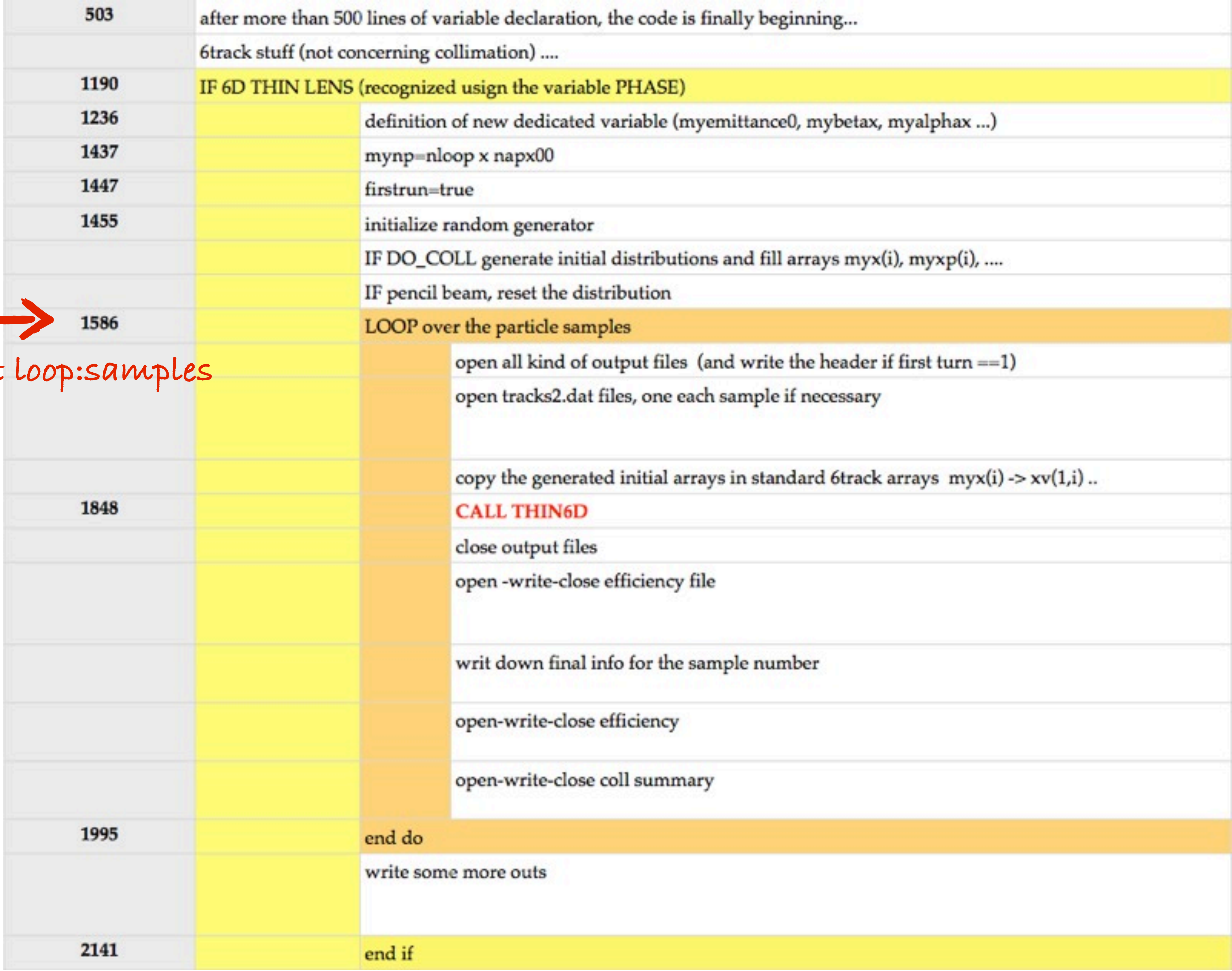


| | | | |
|---|---|---|---|
| 503 | after more than 500 lines of variable declaration, the code is finally beginning... | | |
| | 6track stuff (not concerning collimation) .... | | |
| 1190 | IF 6D THIN LENS (recognized usign the variable PHASE) | | |
| 1236 | | definition of new dedicated variable (myemittance0, mybetax, myalphax ...) | |
| 1437 | | mynp=nloop x napx00 | |
| 1447 | | firstrun=true | |
| 1455 | | initialize random generator | |
| | | IF DO_COLL generate initial distributions and fill arrays myx(i), myxp(i), .... | |
| | | IF pencil beam, reset the distribution | |
| 1586 | | LOOP over the particle samples | |
| | | | open all kind of output files (and write the header if first turn ==1) |
| | | | open tracks2.dat files, one each sample if necessary |
| | | | copy the generated initial arrays in standard 6track arrays myx(i) -> xv(1,i) .. |
| 1848 | | | CALL THIN6D |
| | | | close output files |
| | | | open -write-close efficiency file |
| | | | writ down final info for the sample number |
| | | | open-write-close efficiency |
| | | | open-write-close coll summary |
| 1995 | | end do | |
| | | write some more outs | |
| 2141 | | end if | |

main instructions

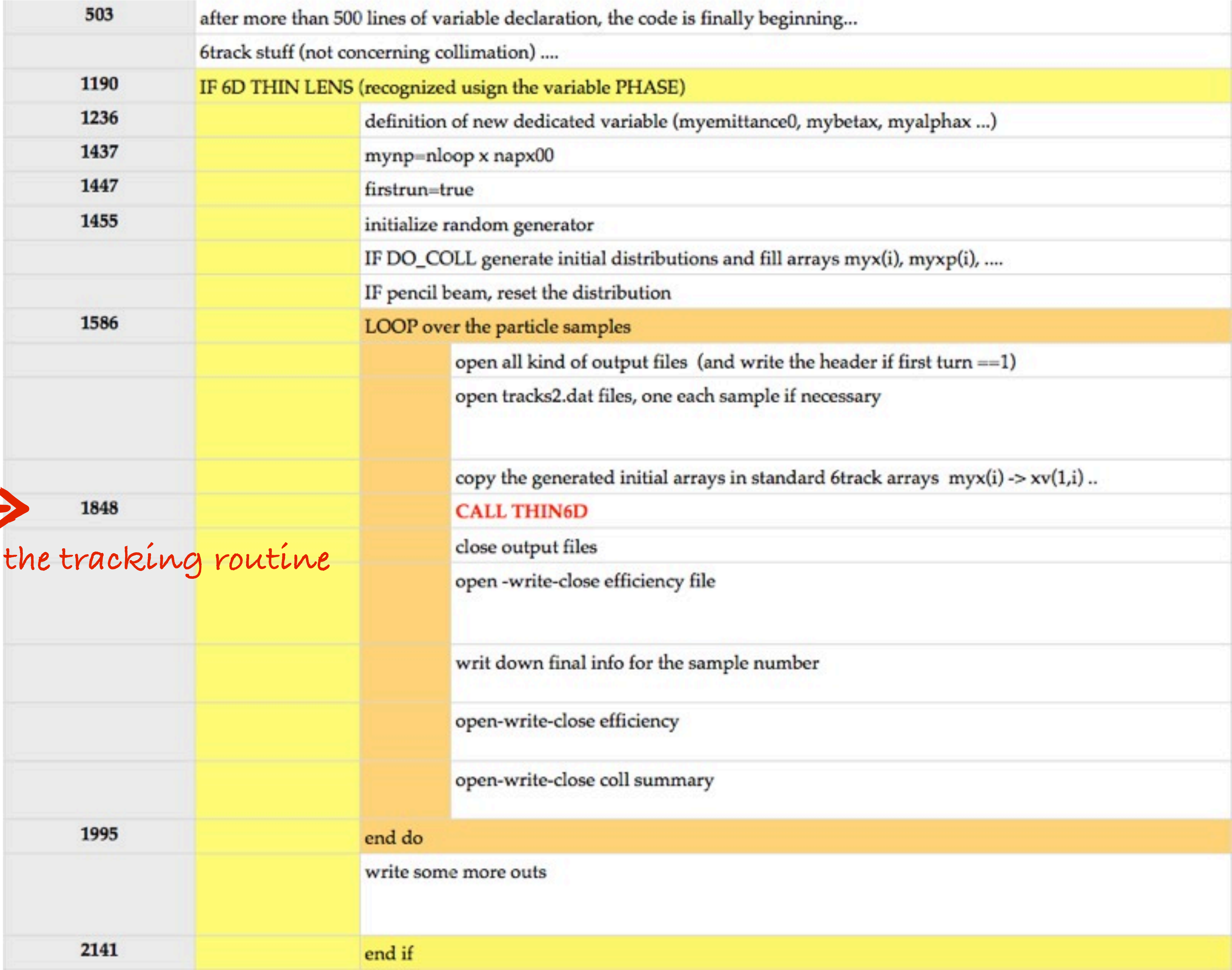
# track.f –> thauthin

first loop:samples



| | | | |
|---|---|---|---|
| 503 | after more than 500 lines of variable declaration, the code is finally beginning... | | |
| | 6track stuff (not concerning collimation) .... | | |
| 1190 | IF 6D THIN LENS (recognized usign the variable PHASE) | | |
| 1236 | | definition of new dedicated variable (myemittance0, mybetax, myalphax ...) | |
| 1437 | | mynp=nloop x napx00 | |
| 1447 | | firstrun=true | |
| 1455 | | initialize random generator | |
| | | IF DO_COLL generate initial distributions and fill arrays myx(i), myxp(i), .... | |
| | | IF pencil beam, reset the distribution | |
| 1586 | | LOOP over the particle samples | |
| | | | open all kind of output files (and write the header if first turn ==1) |
| | | | open tracks2.dat files, one each sample if necessary |
| | | | copy the generated initial arrays in standard 6track arrays myx(i) -> xv(1,i) .. |
| 1848 | | | CALL THIN6D |
| | | | close output files |
| | | | open -write-close efficiency file |
| | | | writ down final info for the sample number |
| | | | open-write-close efficiency |
| | | | open-write-close coll summary |
| 1995 | | end do | |
| | | write some more outs | |
| 2141 | | end if | |

# track.f –> thauthin

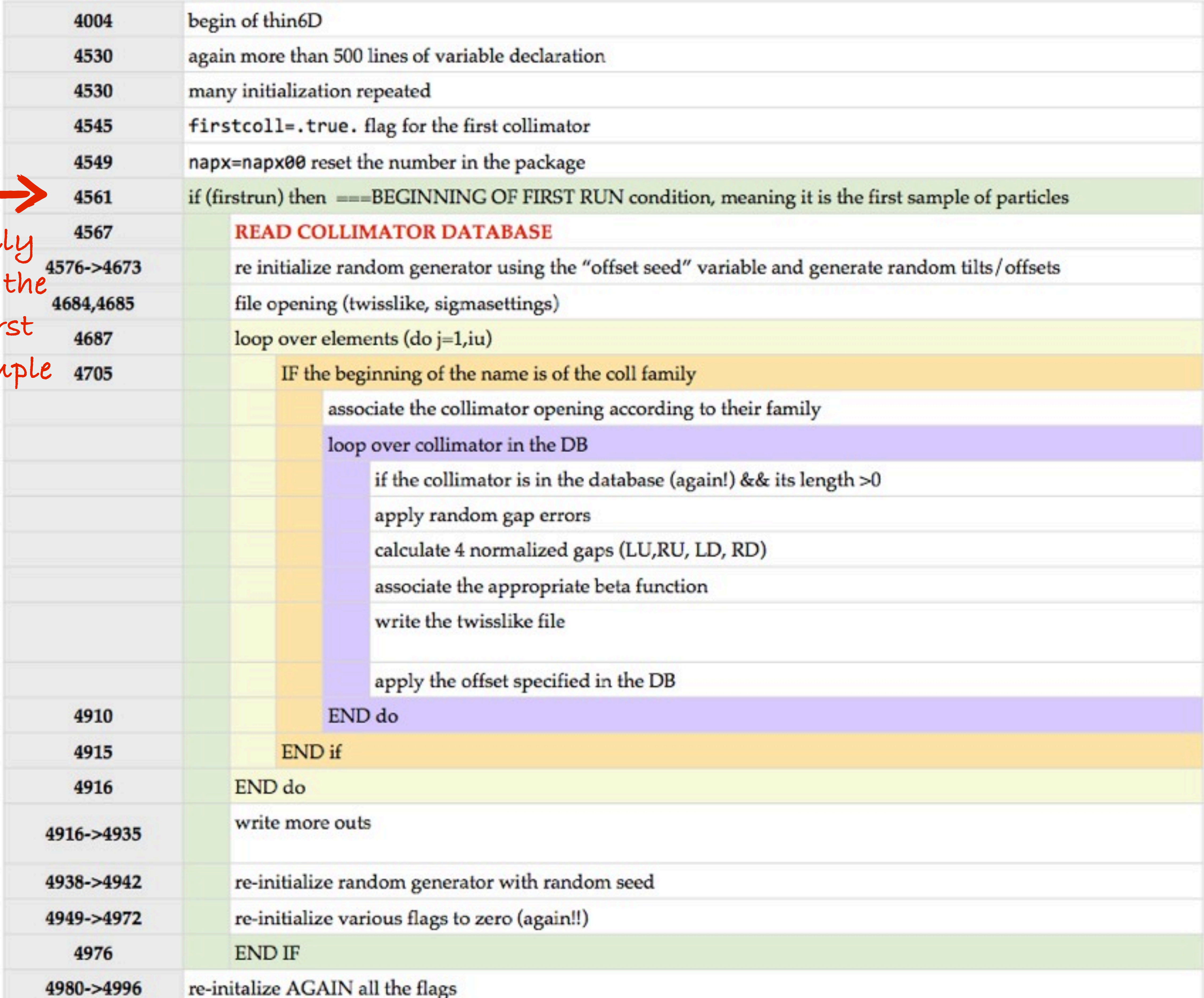


| | | | |
|---|---|---|---|
| 503 | after more than 500 lines of variable declaration, the code is finally beginning... | | |
| | 6track stuff (not concerning collimation) .... | | |
| 1190 | IF 6D THIN LENS (recognized usign the variable PHASE) | | |
| 1236 | | definition of new dedicated variable (myemittance0, mybetax, myalphax ...) | |
| 1437 | | mynp=nloop x napx00 | |
| 1447 | | firstrun=true | |
| 1455 | | initialize random generator | |
| | | IF DO_COLL generate initial distributions and fill arrays myx(i), myxp(i), .... | |
| | | IF pencil beam, reset the distribution | |
| 1586 | | LOOP over the particle samples | |
| | | | open all kind of output files  (and write the header if first turn ==1) |
| | | | open tracks2.dat files, one each sample if necessary |
| | | | copy the generated initial arrays in standard 6track arrays  myx(i) -> xv(1,i) .. |
| 1848 | | | CALL THIN6D |
| | | | close output files |
| | | | open -write-close efficiency file |
| | | | writ down final info for the sample number |
| | | | open-write-close efficiency |
| | | | open-write-close coll summary |
| 1995 | | end do | |
| | | write some more outs | |
| 2141 | | end if | |

call the tracking routine

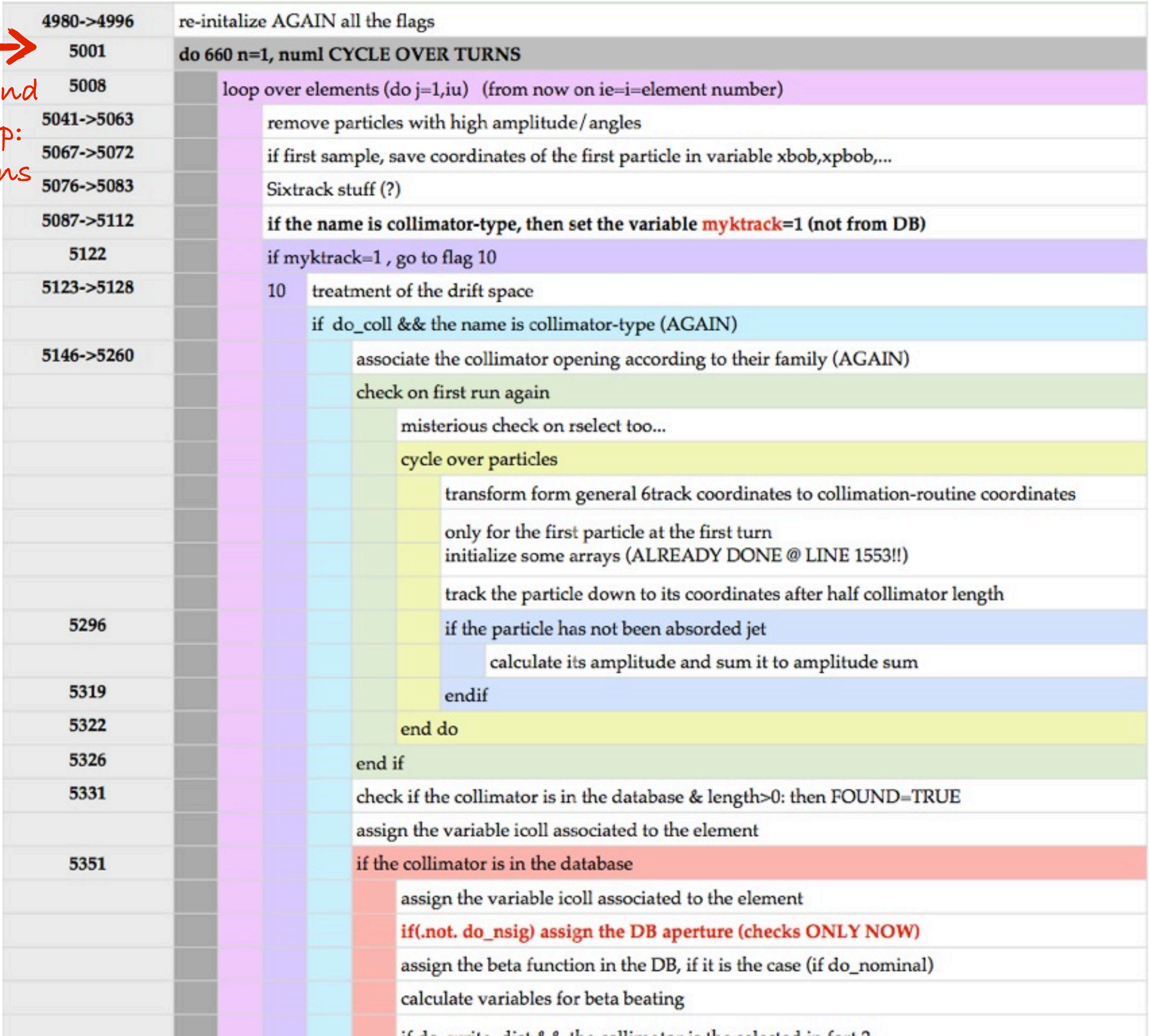
# track.f -> thin6D

only for the first sample



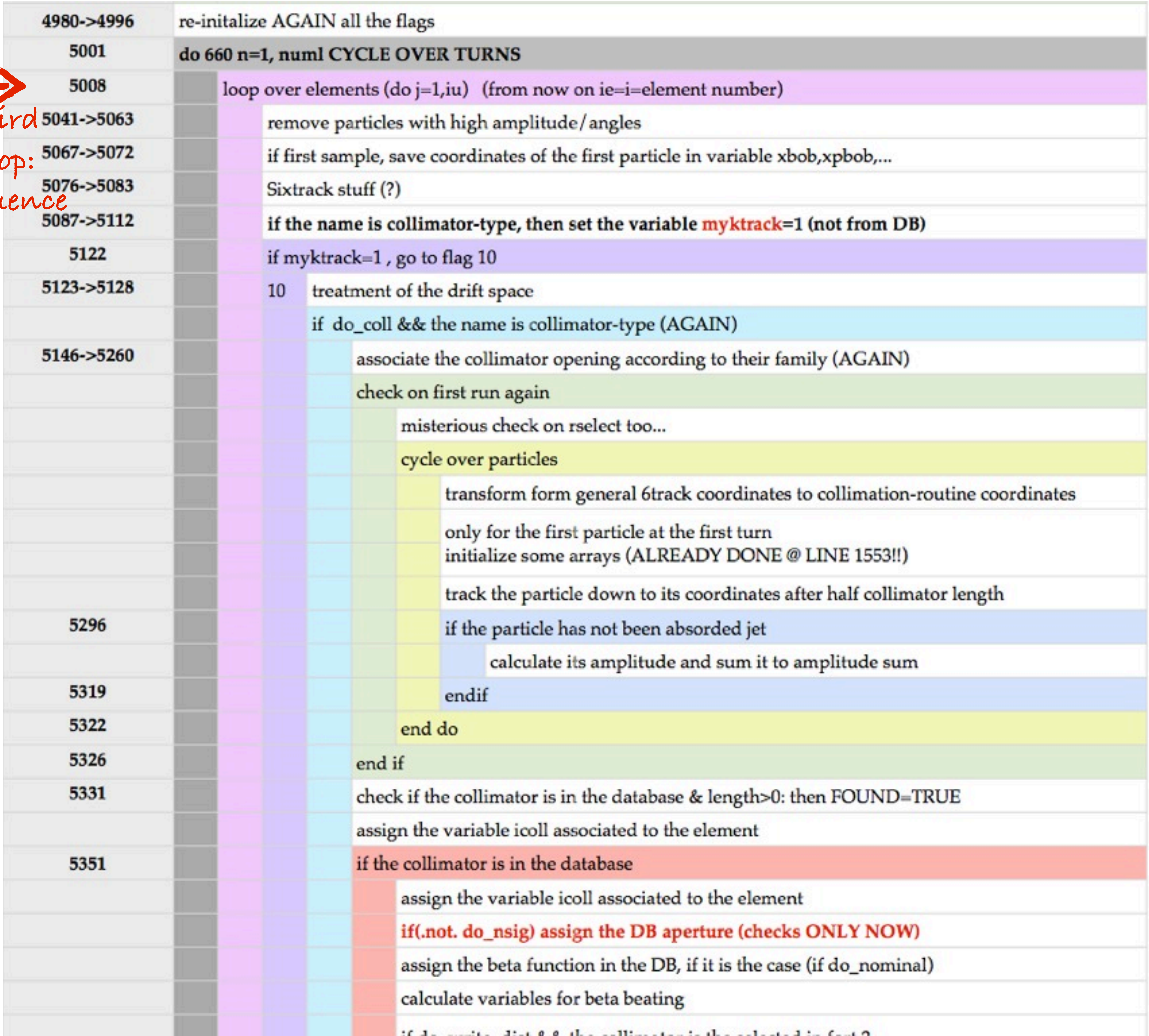
| | | | | | |
|---|---|---|---|---|---|
| 4004 | begin of thin6D | | | | |
| 4530 | again more than 500 lines of variable declaration | | | | |
| 4530 | many initialization repeated | | | | |
| 4545 | `firstcoll=.true.` flag for the first collimator | | | | |
| 4549 | `napx=napx00` reset the number in the package | | | | |
| 4561 | if (firstrun) then ===BEGINNING OF FIRST RUN condition, meaning it is the first sample of particles | | | | |
| 4567 | | READ COLLIMATOR DATABASE | | | |
| 4576->4673 | | re initialize random generator using the "offset seed" variable and generate random tilts/offsets | | | |
| 4684,4685 | | file opening (twisslike, sigmasettings) | | | |
| 4687 | | loop over elements (do j=1,iu) | | | |
| 4705 | | | IF the beginning of the name is of the coll family | | |
| | | | | associate the collimator opening according to their family | |
| | | | | loop over collimator in the DB | |
| | | | | | if the collimator is in the database (again!) && its length >0 |
| | | | | | apply random gap errors |
| | | | | | calculate 4 normalized gaps (LU,RU, LD, RD) |
| | | | | | associate the appropriate beta function |
| | | | | | write the twisslike file |
| | | | | | apply the offset specified in the DB |
| 4910 | | | | END do | |
| 4915 | | | END if | | |
| 4916 | | END do | | | |
| 4916->4935 | | write more outs | | | |
| 4938->4942 | | re-initialize random generator with random seed | | | |
| 4949->4972 | | re-initialize various flags to zero (again!!) | | | |
| 4976 | | END IF | | | |
| 4980->4996 | re-initalize AGAIN all the flags | | | | |

second loop: turns

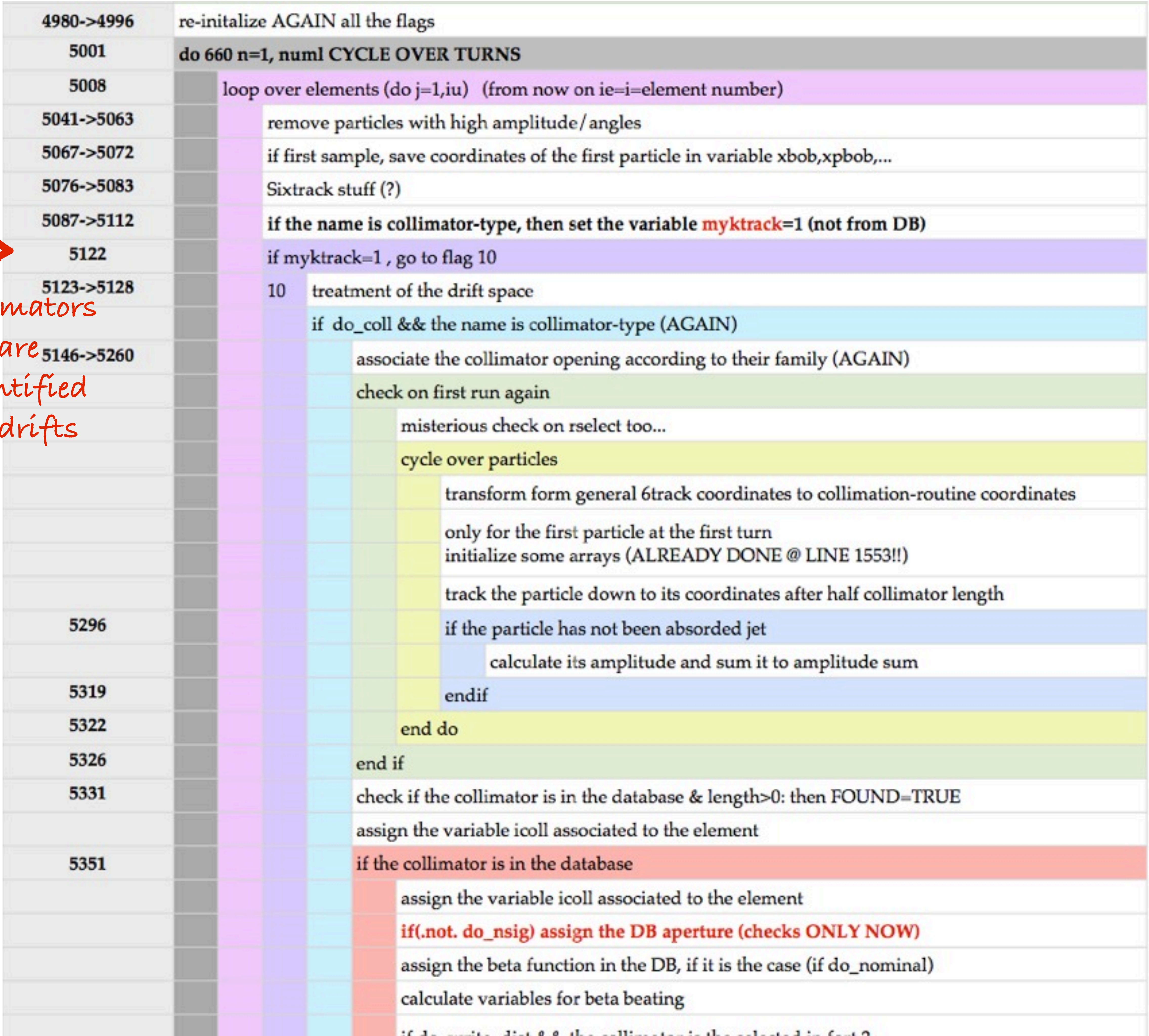


```
4980->4996  re-initalize AGAIN all the flags
5001        do 660 n=1, numl CYCLE OVER TURNS
5008          loop over elements (do j=1,iu)   (from now on ie=i=element number)
5041->5063      remove particles with high amplitude/angles
5067->5072      if first sample, save coordinates of the first particle in variable xbob,xpbob,...
5076->5083      Sixtrack stuff (?)
5087->5112      if the name is collimator-type, then set the variable myktrack=1 (not from DB)
5122            if myktrack=1 , go to flag 10
5123->5128      10    treatment of the drift space
                      if  do_coll && the name is collimator-type (AGAIN)
5146->5260              associate the collimator opening according to their family (AGAIN)
                        check on first run again
                          misterious check on rselect too...
                          cycle over particles
                            transform form general 6track coordinates to collimation-routine coordinates
                            only for the first particle at the first turn
                            initialize some arrays (ALREADY DONE @ LINE 1553!!)
                            track the particle down to its coordinates after half collimator length
5296                        if the particle has not been absorded jet
                              calculate its amplitude and sum it to amplitude sum
5319                        endif
5322                      end do
5326                    end if
5331                    check if the collimator is in the database & length>0: then FOUND=TRUE
                        assign the variable icoll associated to the element
5351                    if the collimator is in the database
                          assign the variable icoll associated to the element
                          if(.not. do_nsig) assign the DB aperture (checks ONLY NOW)
                          assign the beta function in the DB, if it is the case (if do_nominal)
                          calculate variables for beta beating
                          if do_write_dist && the collimator is the selected in fort.3
```

third loop: sequence

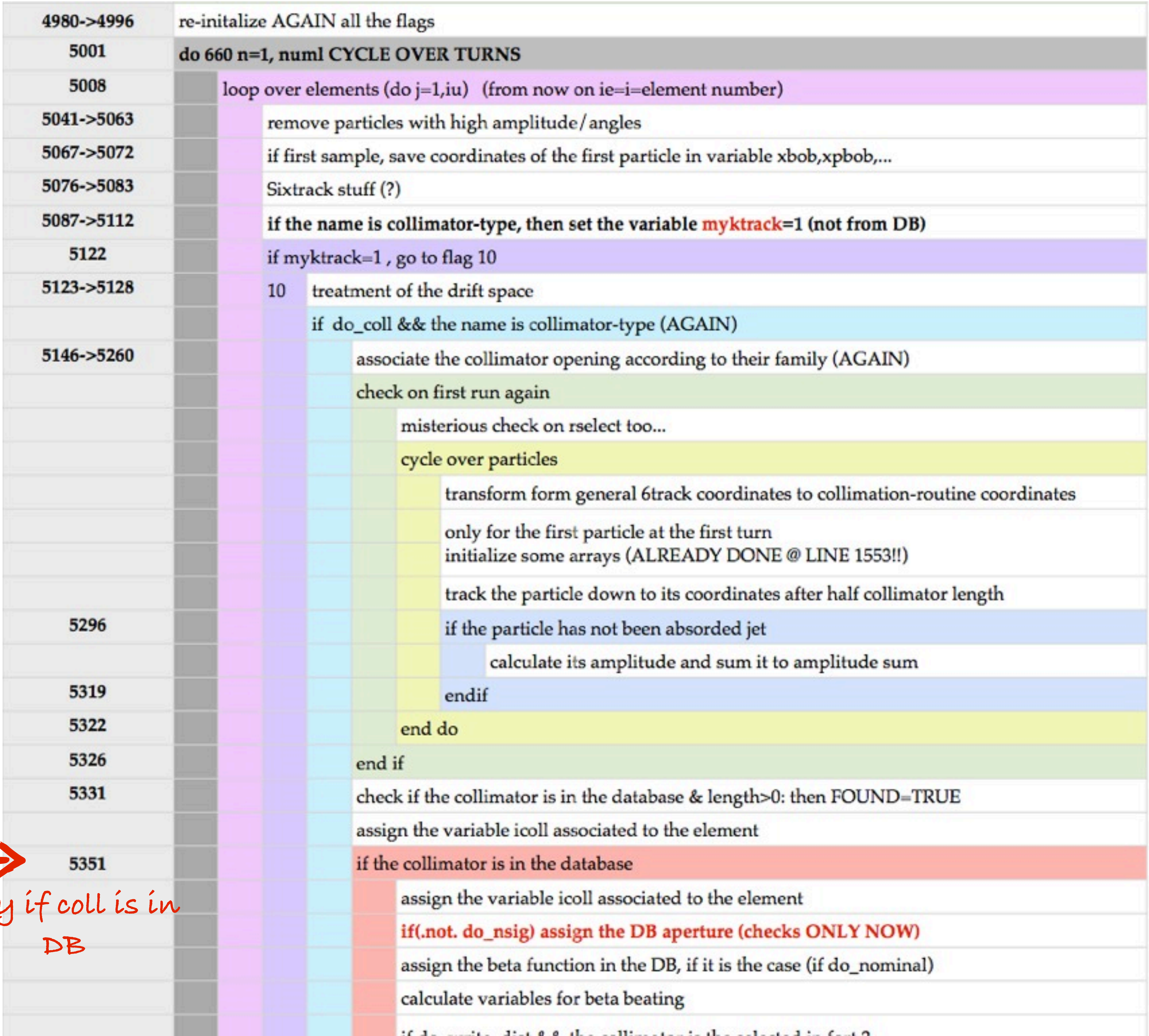


```
4980->4996  re-initalize AGAIN all the flags
5001        do 660 n=1, numl CYCLE OVER TURNS
5008            loop over elements (do j=1,iu)   (from now on ie=i=element number)
5041->5063          remove particles with high amplitude/angles
5067->5072          if first sample, save coordinates of the first particle in variable xbob,xpbob,...
5076->5083          Sixtrack stuff (?)
5087->5112          if the name is collimator-type, then set the variable myktrack=1 (not from DB)
5122                if myktrack=1 , go to flag 10
5123->5128          10  treatment of the drift space
                        if  do_coll && the name is collimator-type (AGAIN)
5146->5260                  associate the collimator opening according to their family (AGAIN)
                            check on first run again
                                misterious check on rselect too...
                                cycle over particles
                                    transform form general 6track coordinates to collimation-routine coordinates
                                    only for the first particle at the first turn
                                    initialize some arrays (ALREADY DONE @ LINE 1553!!)
                                    track the particle down to its coordinates after half collimator length
5296                                if the particle has not been absorded jet
                                        calculate its amplitude and sum it to amplitude sum
5319                                endif
5322                            end do
5326                        end if
5331                        check if the collimator is in the database & length>0: then FOUND=TRUE
                            assign the variable icoll associated to the element
5351                        if the collimator is in the database
                                assign the variable icoll associated to the element
                                if(.not. do_nsig) assign the DB aperture (checks ONLY NOW)
                                assign the beta function in the DB, if it is the case (if do_nominal)
                                calculate variables for beta beating
                                if do_write_dist && the collimator is the selected in fort.3
```



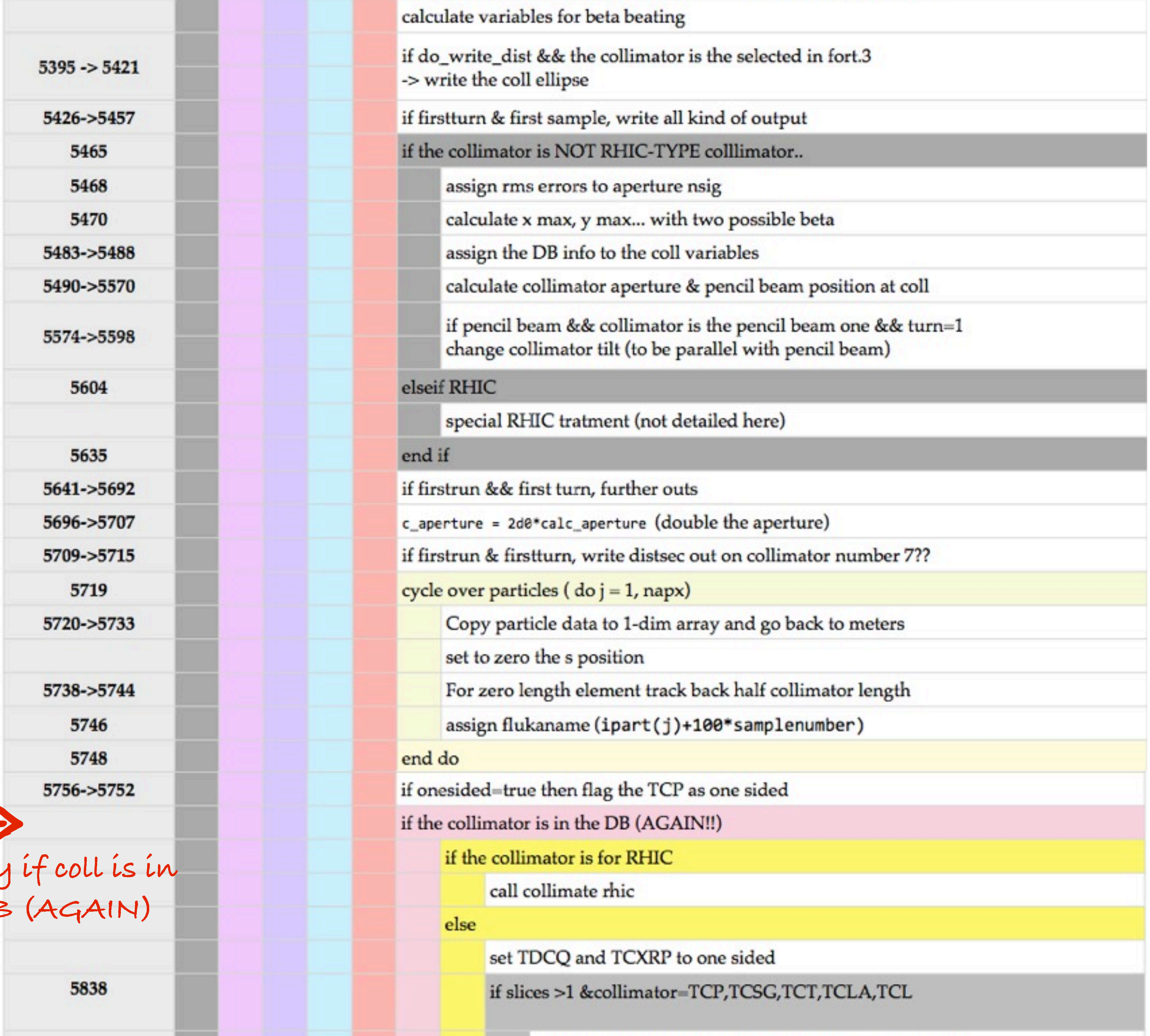
```
4980->4996  re-initalize AGAIN all the flags
5001        do 660 n=1, numl CYCLE OVER TURNS
5008            loop over elements (do j=1,iu)   (from now on ie=i=element number)
5041->5063          remove particles with high amplitude/angles
5067->5072          if first sample, save coordinates of the first particle in variable xbob,xpbob,...
5076->5083          Sixtrack stuff (?)
5087->5112          if the name is collimator-type, then set the variable myktrack=1 (not from DB)
5122                if myktrack=1 , go to flag 10
5123->5128          10  treatment of the drift space
                        if  do_coll && the name is collimator-type (AGAIN)
5146->5260                  associate the collimator opening according to their family (AGAIN)
                            check on first run again
                                misterious check on rselect too...
                                cycle over particles
                                    transform form general 6track coordinates to collimation-routine coordinates
                                    only for the first particle at the first turn
                                    initialize some arrays (ALREADY DONE @ LINE 1553!!)
                                    track the particle down to its coordinates after half collimator length
5296                                if the particle has not been absorded jet
                                        calculate its amplitude and sum it to amplitude sum
5319                                endif
5322                            end do
5326                        end if
5331                        check if the collimator is in the database & length>0: then FOUND=TRUE
                            assign the variable icoll associated to the element
5351                        if the collimator is in the database
                                assign the variable icoll associated to the element
                                if(.not. do_nsig) assign the DB aperture (checks ONLY NOW)
                                assign the beta function in the DB, if it is the case (if do_nominal)
                                calculate variables for beta beating
                                if do_write_dist && the collimator is the selected in fort.3
```

collimators are identified as drifts



```
4980->4996  re-initalize AGAIN all the flags
5001        do 660 n=1, numl CYCLE OVER TURNS
5008            loop over elements (do j=1,iu)   (from now on ie=i=element number)
5041->5063          remove particles with high amplitude/angles
5067->5072          if first sample, save coordinates of the first particle in variable xbob,xpbob,...
5076->5083          Sixtrack stuff (?)
5087->5112          if the name is collimator-type, then set the variable myktrack=1 (not from DB)
5122                if myktrack=1 , go to flag 10
5123->5128          10  treatment of the drift space
                        if  do_coll && the name is collimator-type (AGAIN)
5146->5260                  associate the collimator opening according to their family (AGAIN)
                            check on first run again
                                misterious check on rselect too...
                                cycle over particles
                                    transform form general 6track coordinates to collimation-routine coordinates
                                    only for the first particle at the first turn
                                    initialize some arrays (ALREADY DONE @ LINE 1553!!)
                                    track the particle down to its coordinates after half collimator length
5296                                if the particle has not been absorded jet
                                        calculate its amplitude and sum it to amplitude sum
5319                                endif
5322                            end do
5326                        end if
5331                        check if the collimator is in the database & length>0: then FOUND=TRUE
                            assign the variable icoll associated to the element
5351                        if the collimator is in the database
                                assign the variable icoll associated to the element
                                if(.not. do_nsig) assign the DB aperture (checks ONLY NOW)
                                assign the beta function in the DB, if it is the case (if do_nominal)
                                calculate variables for beta beating
                                if do_write_dist && the collimator is the selected in fort.3
```

→ only if coll is in DB



| | | | |
|---|---|---|---|
| | calculate variables for beta beating | | |
| 5395 -> 5421 | if do_write_dist && the collimator is the selected in fort.3<br>-> write the coll ellipse | | |
| 5426->5457 | if firstturn & first sample, write all kind of output | | |
| 5465 | if the collimator is NOT RHIC-TYPE colllimator.. | | |
| 5468 | | assign rms errors to aperture nsig | |
| 5470 | | calculate x max, y max... with two possible beta | |
| 5483->5488 | | assign the DB info to the coll variables | |
| 5490->5570 | | calculate collimator aperture & pencil beam position at coll | |
| 5574->5598 | | if pencil beam && collimator is the pencil beam one && turn=1<br>change collimator tilt (to be parallel with pencil beam) | |
| 5604 | elseif RHIC | | |
| | | special RHIC tratment (not detailed here) | |
| 5635 | end if | | |
| 5641->5692 | if firstrun && first turn, further outs | | |
| 5696->5707 | `c_aperture = 2d0*calc_aperture` (double the aperture) | | |
| 5709->5715 | if firstrun & firstturn, write distsec out on collimator number 7?? | | |
| 5719 | cycle over particles ( do j = 1, napx) | | |
| 5720->5733 | | Copy particle data to 1-dim array and go back to meters | |
| | | set to zero the s position | |
| 5738->5744 | | For zero length element track back half collimator length | |
| 5746 | | assign flukaname (`ipart(j)+100*samplenumber`) | |
| 5748 | end do | | |
| 5756->5752 | if onesided=true then flag the TCP as one sided | | |
| | if the collimator is in the DB (AGAIN!!) | | |
| | | if the collimator is for RHIC | |
| | | | call collimate rhic |
| | | else | |
| | | | set TDCQ and TCXRP to one sided |
| 5838 | | | if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL |

only if coll is in DB (AGAIN)

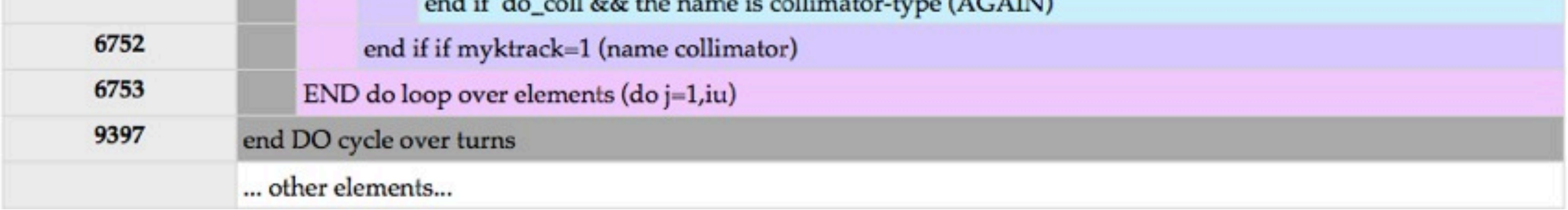
```
5838                if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
                       ugly series of do cycles (9X)-could be easily merged
5997->6085             for each slice, call COLLIMATE2
6086                else
6088                   call COLLIMATE2
physics! 6094       end if (slices)
6097             end if collimator for RHIC
6099          end if (collimator in DB)
6122->6126    initialize impact variables
              cycle over number of particles (do j = 1, napx)
                 IF particle hit : part_hit(j).eq.(10000*ie+iturn))
                    For zero length element track back half collimator length
                    copy data back to original verctor (S IS NOT TOUCHED)
6160->6168          Energy update
                 else
                    copy back the initial coordinates (necessary??)
6178             end if particle hit
                 --- commented code---
6251             IF particle hit : part_hit(j).eq.(10000*ie+iturn))
                    write impacts, if flag is on (do_write_impacts)
                    if particle is absorbed & wirte impacts, another file
6266                if particle is absorbed, write tracks2.dat
                    if the particle has not been absorbed
                       calculate kick
                       assign to adeguate halo family
                    end if particle not absorbed
6319                if  dowritetracks
                       if particel not absorbed
6326                      if particle in some halo & coordinates < 99 and normalized positions
```

```
5838        if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
                ugly series of do cycles (9X)-could be easily merged
5997->6085      for each slice, call COLLIMATE2
6086        else
6088            call COLLIMATE2
6094        end if (slices)
6097    end if collimator for RHIC
6099 end if (collimator in DB)
6122->6126 initialize impact variables
```

processing the results

```
     cycle over number of particles (do j = 1, napx)
        IF particle hit : part_hit(j).eq.(10000*ie+iturn))
            For zero length element track back half collimator length
            copy data back to original verctor (S IS NOT TOUCHED)
6160->6168  Energy update
        else
            copy back the initial coordinates (necessary??)
6178    end if particle hit
        --- commented code---
6251    IF particle hit : part_hit(j).eq.(10000*ie+iturn))
            write impacts, if flag is on (do_write_impacts)
            if particle is absorbed & wirte impacts, another file
6266        if particle is absorbed, write tracks2.dat
            if the particle has not been absorbed
                calculate kick
                assign to adeguate halo family
            end if particle not absorbed
6319        if  dowritetracks
                if particel not absorbed
6326                if particle in some halo & coordinates < 99 and normalized positions
```

```
                                        if particel not absorbed
6326                                        if particle in some halo & coordinates < 99 and normalized positions are < some cuts
                                                ri-transform coordinates in m,rad
                                                write tracks.dat
                                                write tracks.dat
6369                                        end if cuts
6387                                    end if not absorbed
6389                                end if particle not absorbed
                                    fill histo variables (number of impacts, average...)
6409                                if particle absorbed
                                        increase absorbed number
6420                                    adjust some flags
6421                                end if particle absorbed
6425                            end if particle hit
6431                        END cycle over number of particles
6435->6455                  Calculate statistical observables and save into files
6469                        IF THE COLL is the selected one
                                reset counters for selected collimator
6475->6497                      cycle over particles and update selected collimator counters
6502->6529                      Calculate average impact parameter and save distribution into file
6548                        END IF selected COLL
6645                    end if (collimator in DB)
                    else (if do_coll is false, or not collimator name)
                        drift treatment... (not detailed here)
                    end if  do_coll && the name is collimator-type (AGAIN)
6752            end if if myktrack=1 (name collimator)
6753        END do loop over elements (do j=1,iu)
```

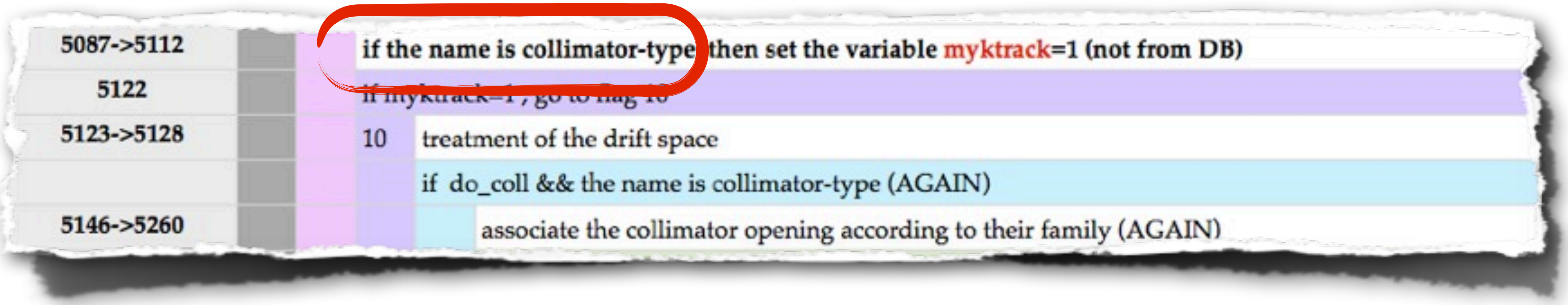


| | |
|---|---|
| | end if do_coll && the name is collimator-type (AGAIN) |
| 6752 | end if if myktrack=1 (name collimator) |
| 6753 | END do loop over elements (do j=1,iu) |
| 9397 | end DO cycle over turns |
| | ... other elements... |

all this is very confusing
(even in this form, imagine reading in Fortran)
the xls file has been put on the web for you to modify, update and complete (see documentation file to this talk in indico)

# new features

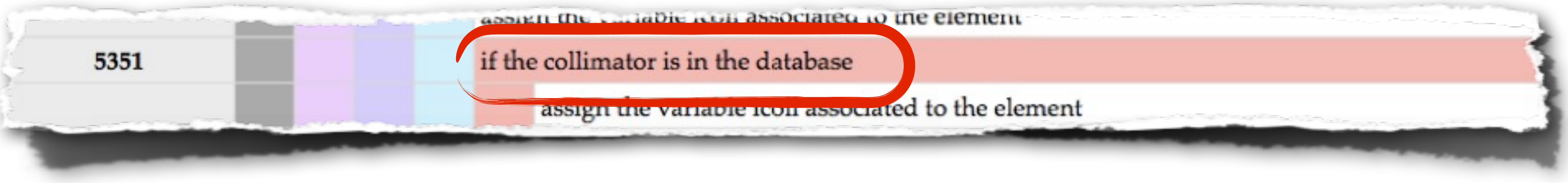
# how2 add a new collimator?



| | | | |
|---|---|---|---|
| 5087->5112 | if the name is collimator-type then set the variable myktrack=1 (not from DB) | | |
| 5122 | if myktrack=1 , go to flag 10 | | |
| 5123->5128 | 10 | treatment of the drift space | |
| | | if do_coll && the name is collimator-type (AGAIN) | |
| 5146->5260 | | | associate the collimator opening according to their family (AGAIN) |

- the first check is done on the collimator name. The new collimator must be recognizable by its name
    - e-lens elements MUST be called elens*
    - tune modulation elements must be called either TM_DIP* or TM_QUAD*
- **this check is performed MANY TIMES in the track.f, watch out! the new name must be included every time**

# how2 add a new collimator?



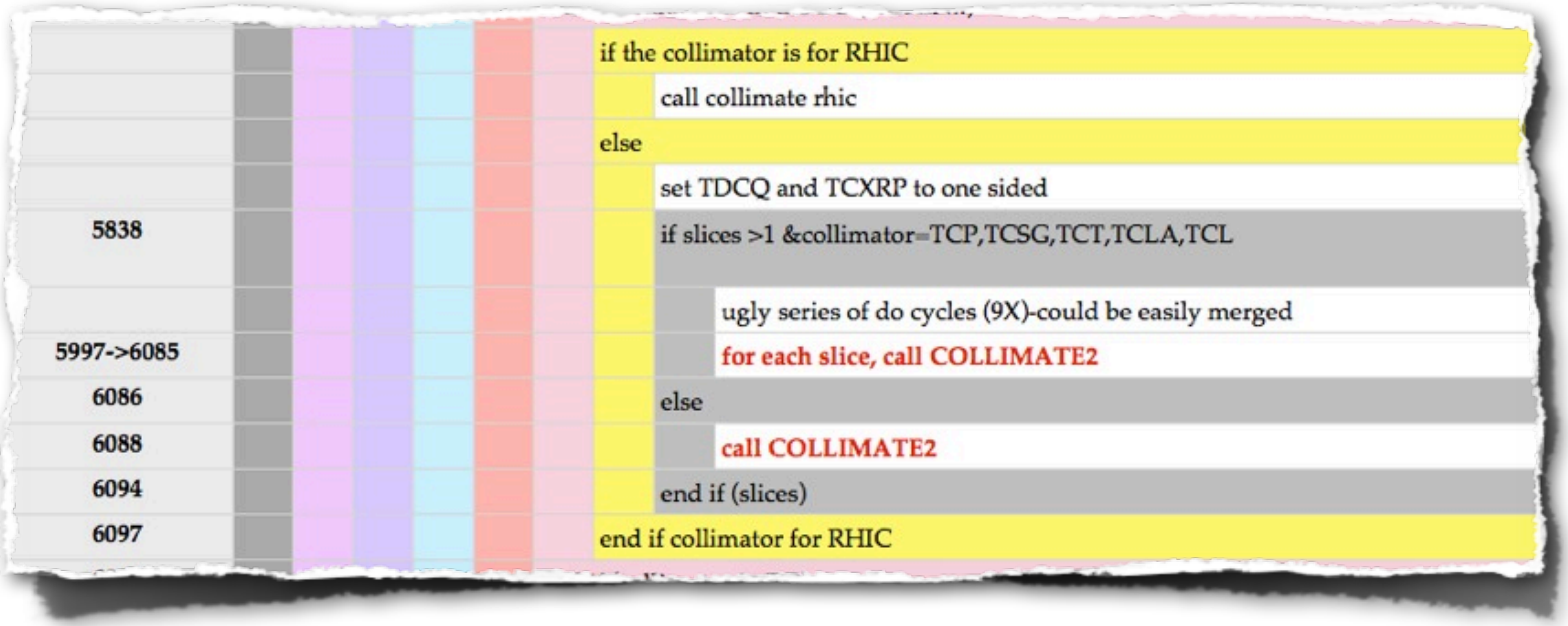
assign the variable icon associated to the element

5351 if the collimator is in the database

assign the variable icon associated to the element

- the new collimator must be included in the database
- each element has specific characteristics => for each element **a new database entry must be created**
- **README files are available for full DB element description**
- the **sixve.f** (file where the DB is read) **must be modified** accordingly

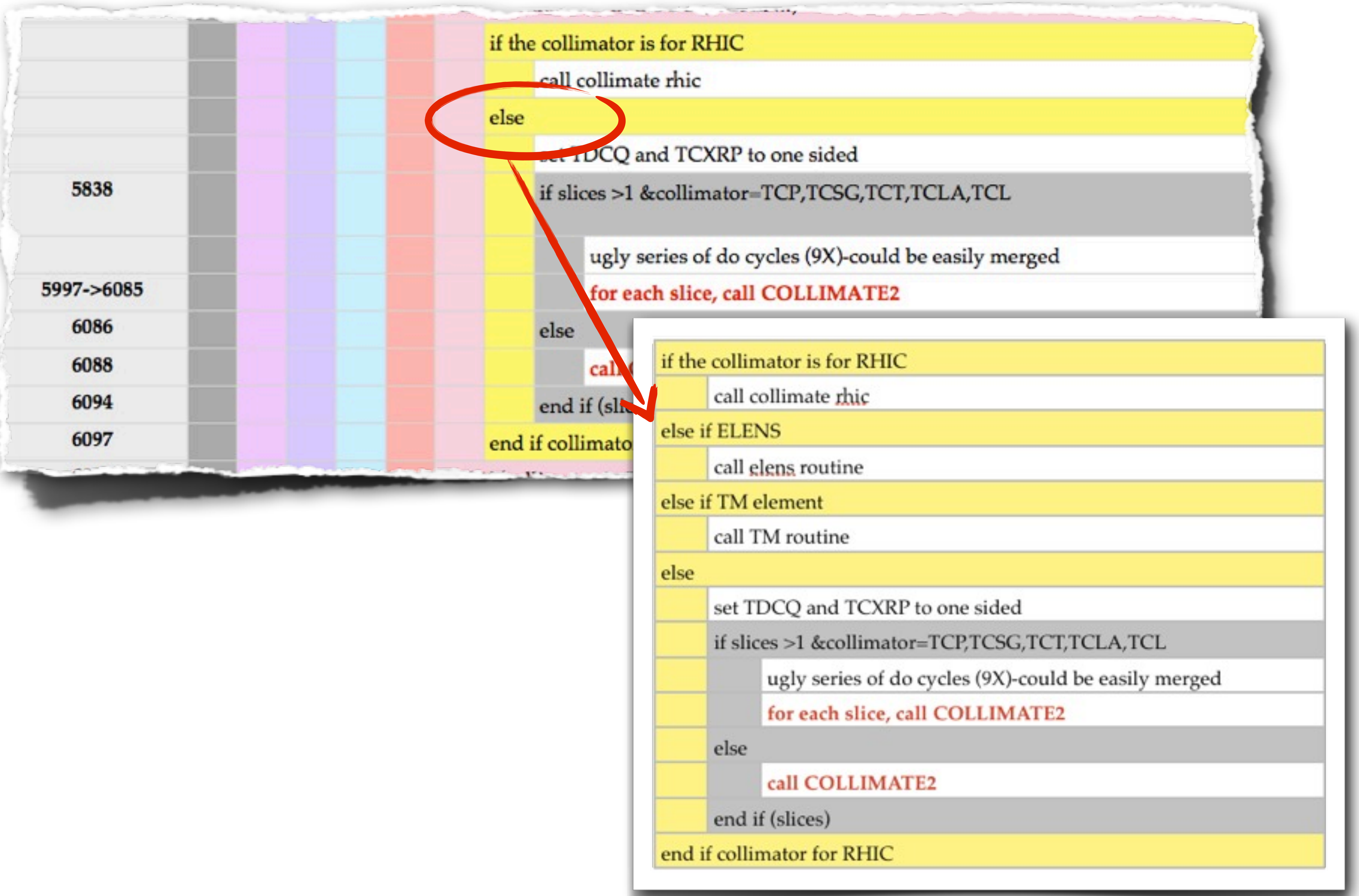
# how2 add a new collimator?



```
             if the collimator is for RHIC
                 call collimate rhic
             else
                 set TDCQ and TCXRP to one sided
5838             if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
                     ugly series of do cycles (9X)-could be easily merged
5997->6085           for each slice, call COLLIMATE2
6086             else
6088                 call COLLIMATE2
6094             end if (slices)
6097         end if collimator for RHIC
```

- obviously, a new collimator is inserted because it acts differently on the particles
- if the new element is found, **a new physics routine must be called**
- the scattering routine for elens and for modulation elements are written in separte file, respectively elense.f and tune_mod.f

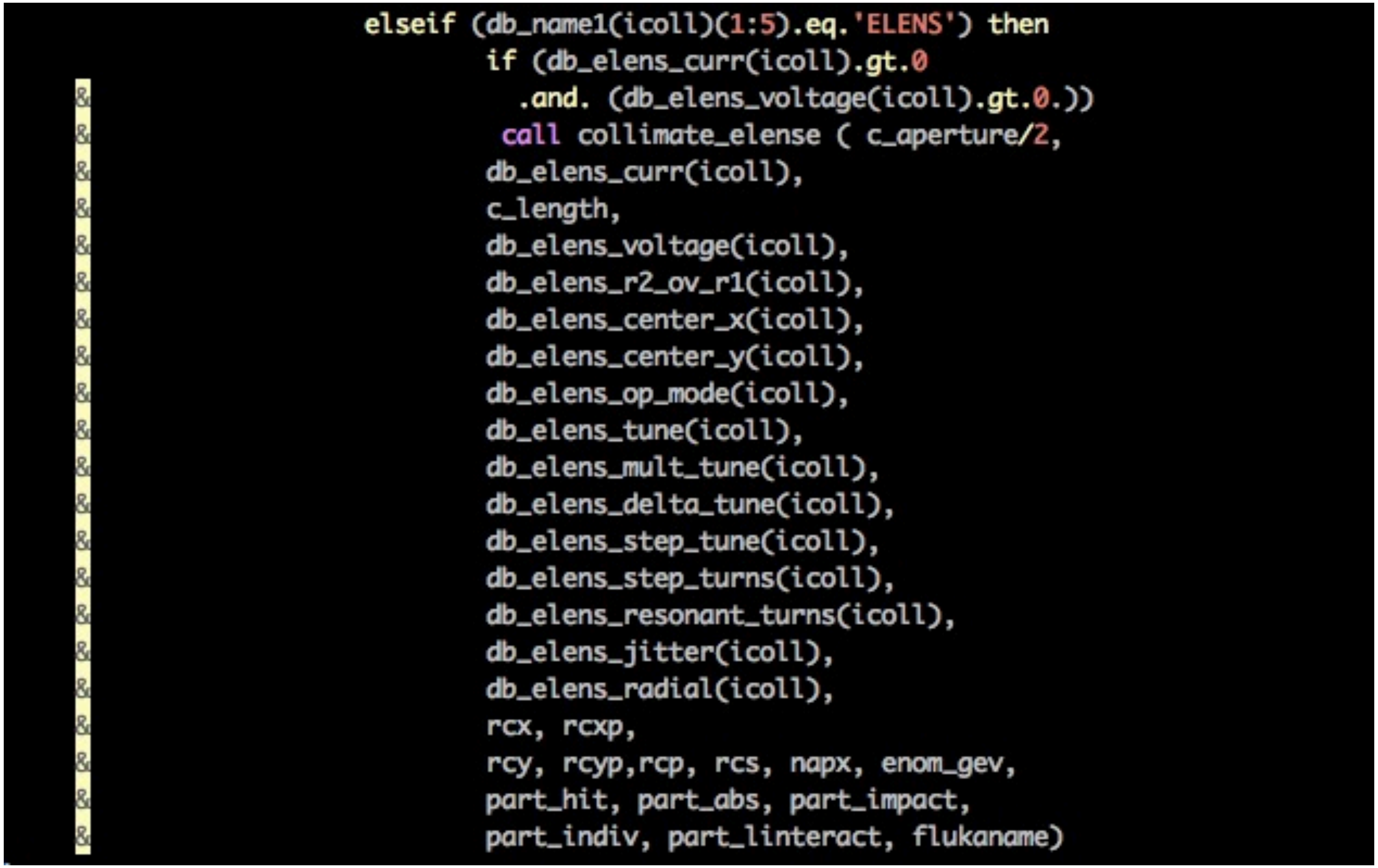
# how2 add a new collimator?



```
            if the collimator is for RHIC
                call collimate rhic
            else
                set TDCQ and TCXRP to one sided
5838            if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
                    ugly series of do cycles (9X)-could be easily merged
5997->6085          for each slice, call COLLIMATE2
6086            else
6088                cal
6094            end if (sl
6097        end if collimato
```

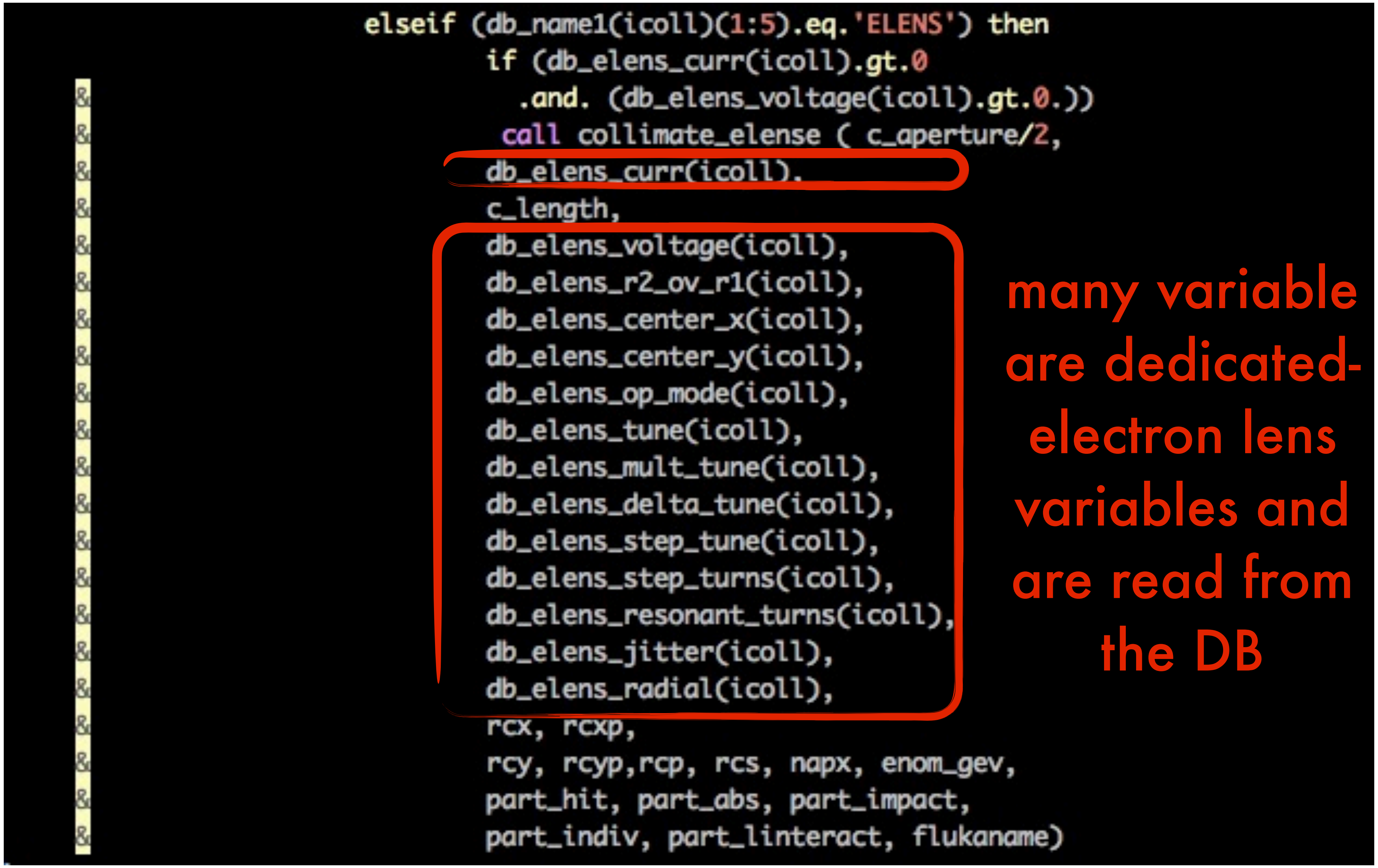
```
if the collimator is for RHIC
    call collimate rhic
else if ELENS
    call elens routine
else if TM element
    call TM routine
else
    set TDCQ and TCXRP to one sided
    if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
        ugly series of do cycles (9X)-could be easily merged
        for each slice, call COLLIMATE2
    else
        call COLLIMATE2
    end if (slices)
end if collimator for RHIC
```

# Elens routine



```
                       elseif (db_name1(icoll)(1:5).eq.'ELENS') then
                               if (db_elens_curr(icoll).gt.0
     &                           .and. (db_elens_voltage(icoll).gt.0.))
     &                          call collimate_elense ( c_aperture/2,
     &                         db_elens_curr(icoll),
     &                         c_length,
     &                         db_elens_voltage(icoll),
     &                         db_elens_r2_ov_r1(icoll),
     &                         db_elens_center_x(icoll),
     &                         db_elens_center_y(icoll),
     &                         db_elens_op_mode(icoll),
     &                         db_elens_tune(icoll),
     &                         db_elens_mult_tune(icoll),
     &                         db_elens_delta_tune(icoll),
     &                         db_elens_step_tune(icoll),
     &                         db_elens_step_turns(icoll),
     &                         db_elens_resonant_turns(icoll),
     &                         db_elens_jitter(icoll),
     &                         db_elens_radial(icoll),
     &                         rcx, rcxp,
     &                         rcy, rcyp,rcp, rcs, napx, enom_gev,
     &                         part_hit, part_abs, part_impact,
     &                         part_indiv, part_linteract, flukaname)
```

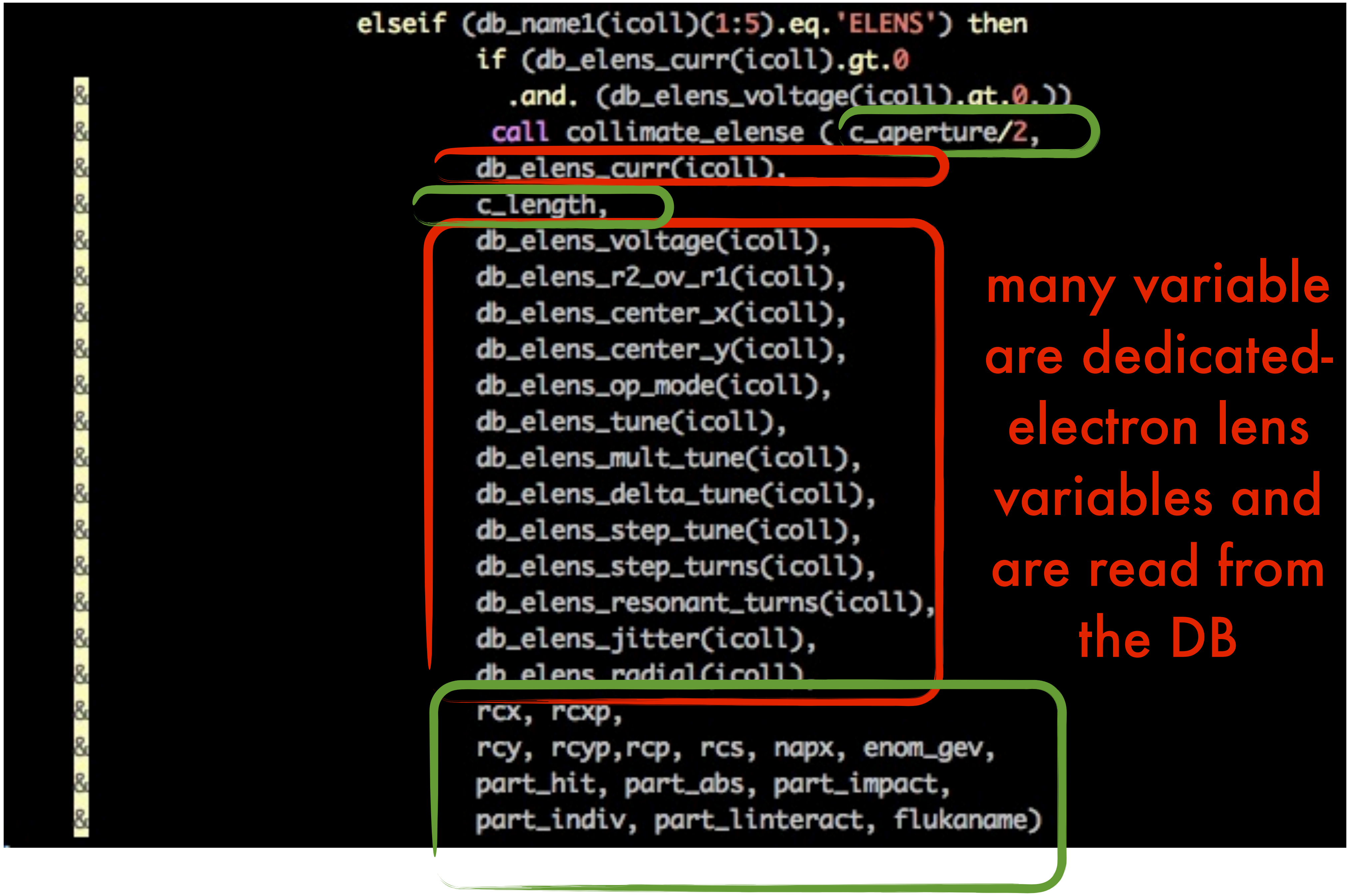
# Elens routine



```
                        elseif (db_name1(icoll)(1:5).eq.'ELENS') then
                                if (db_elens_curr(icoll).gt.0
     &                            .and. (db_elens_voltage(icoll).gt.0.))
     &                           call collimate_elense ( c_aperture/2,
     &                          db_elens_curr(icoll),
     &                          c_length,
     &                          db_elens_voltage(icoll),
     &                          db_elens_r2_ov_r1(icoll),
     &                          db_elens_center_x(icoll),
     &                          db_elens_center_y(icoll),
     &                          db_elens_op_mode(icoll),
     &                          db_elens_tune(icoll),
     &                          db_elens_mult_tune(icoll),
     &                          db_elens_delta_tune(icoll),
     &                          db_elens_step_tune(icoll),
     &                          db_elens_step_turns(icoll),
     &                          db_elens_resonant_turns(icoll),
     &                          db_elens_jitter(icoll),
     &                          db_elens_radial(icoll),
     &                          rcx, rcxp,
     &                          rcy, rcyp,rcp, rcs, napx, enom_gev,
     &                          part_hit, part_abs, part_impact,
     &                          part_indiv, part_linteract, flukaname)
```

many variable are dedicated-electron lens variables and are read from the DB

# Elens routine



```
                 elseif (db_name1(icoll)(1:5).eq.'ELENS') then
                        if (db_elens_curr(icoll).gt.0
     &                    .and. (db_elens_voltage(icoll).gt.0.))
                         call collimate_elense ( c_aperture/2,
     &                  db_elens_curr(icoll),
     &                  c_length,
     &                  db_elens_voltage(icoll),
     &                  db_elens_r2_ov_r1(icoll),
     &                  db_elens_center_x(icoll),
     &                  db_elens_center_y(icoll),
     &                  db_elens_op_mode(icoll),
     &                  db_elens_tune(icoll),
     &                  db_elens_mult_tune(icoll),
     &                  db_elens_delta_tune(icoll),
     &                  db_elens_step_tune(icoll),
     &                  db_elens_step_turns(icoll),
     &                  db_elens_resonant_turns(icoll),
     &                  db_elens_jitter(icoll),
     &                  db_elens_radial(icoll),
     &                  rcx, rcxp,
     &                  rcy, rcyp,rcp, rcs, napx, enom_gev,
     &                  part_hit, part_abs, part_impact,
     &                  part_indiv, part_linteract, flukaname)
```

many variable are dedicated-electron lens variables and are read from the DB

many other variables are common to the standard collimator case and must be checked that they work properly for the elens

# what happens inside?

## Elens routine

- cycle over particle
- transform coordinates in radius, angle
- check if the radius is larger than elens radius
- if this is the case, call the “elens kick” routine
- transform back the coordinate
- close

# what happens inside?

the kick is a radial kick which is determined by the particle position, by the e-lens characteristics and by the operation mode. 3 operation modes are possible: DC, AC, random. A document is being prepared with all the details.

n radius, angle

ger than elens radius

e "elens kick" routine

dinate

- close

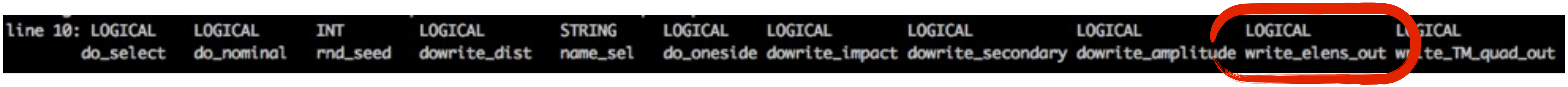
# new outputs

- *(UNIT=887,FORM='UNFORMATTED', file="**elens.bin**")*
  # 1=ncoll 2=npart 3=nturn 4=x0 5=xp0 6=y0 7=yp0 8=kx 9=ky 10=rkick"

- *(UNIT=888,FORM='UNFORMATTED',file="**elens.norm.bin**")*
  "# 1=sample 2=npart 3=nturn 4=xn0 5=xpn0 6=yn0 7=ypn0 8=DeltaAx 13=DeltaAy 14=Ax 15=Ay"

- binary files which are read through dedicated fortran programs (all available)

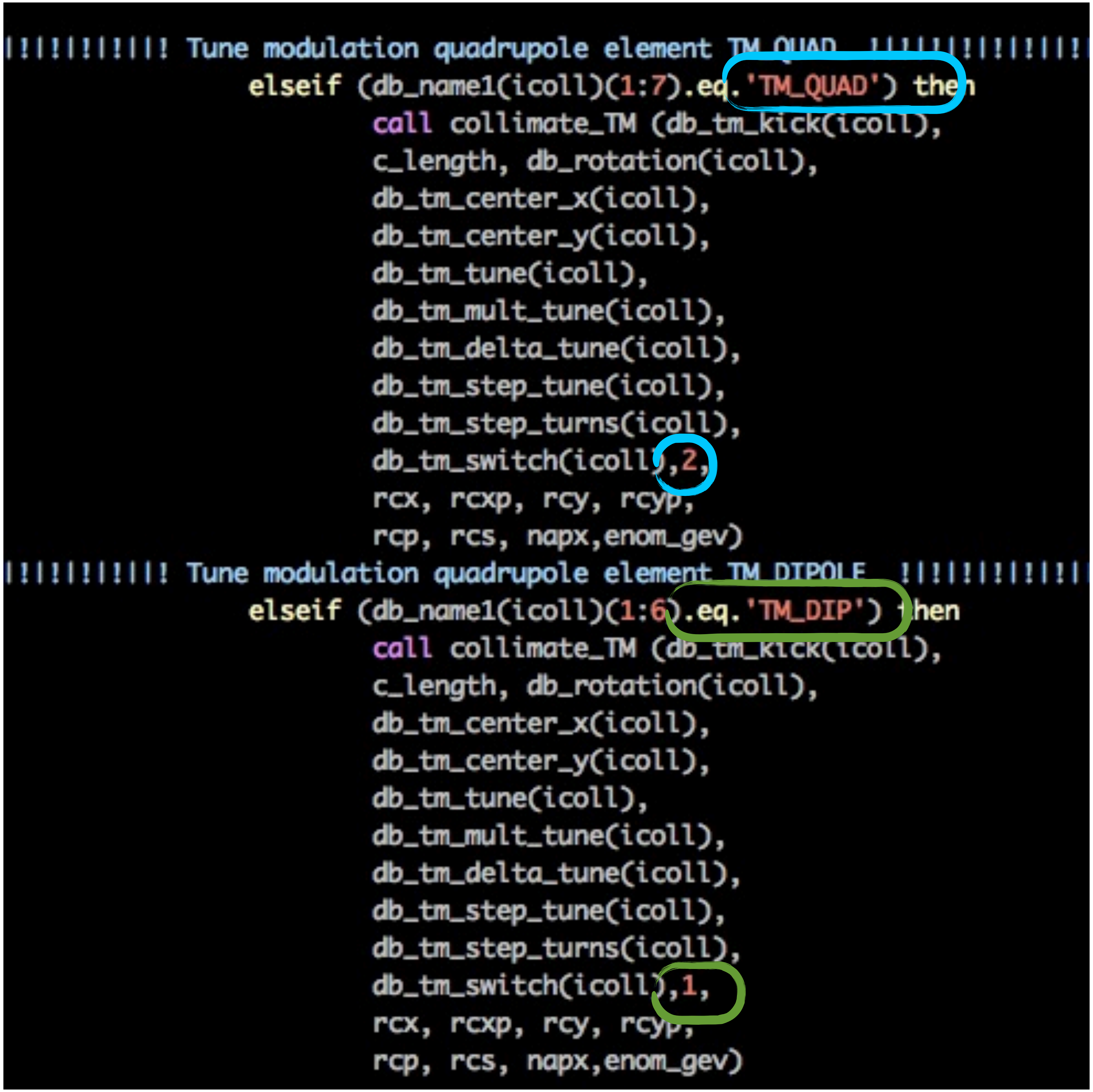
write elens flag!



```
line 10: LOGICAL    LOGICAL     INT       LOGICAL        STRING    LOGICAL     LOGICAL          LOGICAL            LOGICAL            LOGICAL          LOGICAL
         do_select  do_nominal  rnd_seed  dowrite_dist   name_sel  do_oneside  dowrite_impact   dowrite_secondary  dowrite_amplitude  write_elens_out  write_TM_quad_out
```

- actived/deactivated through the "write_elens_out" FLAG read in the fort.3

# Tune modulation routine



```
!!!!!!!!!!!! Tune modulation quadrupole element TM_QUAD  !!!!!!!!!!!!!!
                elseif (db_name1(icoll)(1:7).eq.'TM_QUAD') then
                        call collimate_TM (db_tm_kick(icoll),
                        c_length, db_rotation(icoll),
                        db_tm_center_x(icoll),
                        db_tm_center_y(icoll),
                        db_tm_tune(icoll),
                        db_tm_mult_tune(icoll),
                        db_tm_delta_tune(icoll),
                        db_tm_step_tune(icoll),
                        db_tm_step_turns(icoll),
                        db_tm_switch(icoll),2,
                        rcx, rcxp, rcy, rcyp,
                        rcp, rcs, napx,enom_gev)
!!!!!!!!!!!! Tune modulation quadrupole element TM_DIPOLE  !!!!!!!!!!!!
                elseif (db_name1(icoll)(1:6).eq.'TM_DIP') then
                        call collimate_TM (db_tm_kick(icoll),
                        c_length, db_rotation(icoll),
                        db_tm_center_x(icoll),
                        db_tm_center_y(icoll),
                        db_tm_tune(icoll),
                        db_tm_mult_tune(icoll),
                        db_tm_delta_tune(icoll),
                        db_tm_step_tune(icoll),
                        db_tm_step_turns(icoll),
                        db_tm_switch(icoll),1,
                        rcx, rcxp, rcy, rcyp,
                        rcp, rcs, napx,enom_gev)
```

two tune modulation elements are possible: dipoles and quadrupoles. They are selected in the thin6d according to their name. The routine called is the same, but with a flag “2” for quadrupoles and “1” for the dipole

The database type is also the same.

# what happens inside?

## Tune modulation routine

- cycle over particle
- rotate the coordinates element rotation
- call the “TM kick” rou
- transform back the co
- close

the kick is either a quadrupole or a dipole kick depending on the element type. The kick is pulsed. The pulsing frequency depends on the parameters given in the database and the element type.

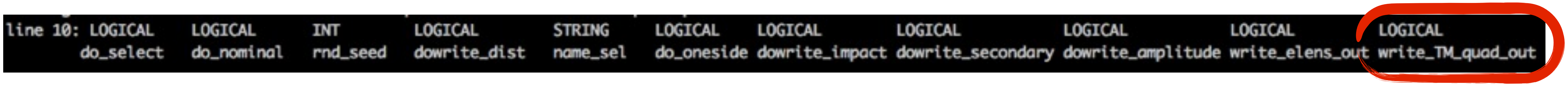
# new outputs

- *(UNIT=889,FORM='UNFORMATTED', file="**tm.bin**")*
  # 1=ncoll 2=npart 3=nturn 4=x0 5=xp0 6=y0 7=yp0 8=kx 9=ky 10=rkick"

- *(UNIT=890,FORM='UNFORMATTED',file="**tm.norm.bin**")*
  "# 1=sample 2=npart 3=nturn 4=xn0 5=xpn0 6=yn0 7=ypn0 8=DAx 13=DAy 14=Ax 15=Ay"

- binary files which are read through dedicated fortran programs (all available)

write tm flag!



```
line 10: LOGICAL    LOGICAL     INT       LOGICAL        STRING     LOGICAL    LOGICAL         LOGICAL            LOGICAL            LOGICAL         LOGICAL
         do_select  do_nominal  rnd_seed  dowrite_dist   name_sel   do_oneside dowrite_impact  dowrite_secondary  dowrite_amplitude  write_elens_out write_TM_quad_out
```

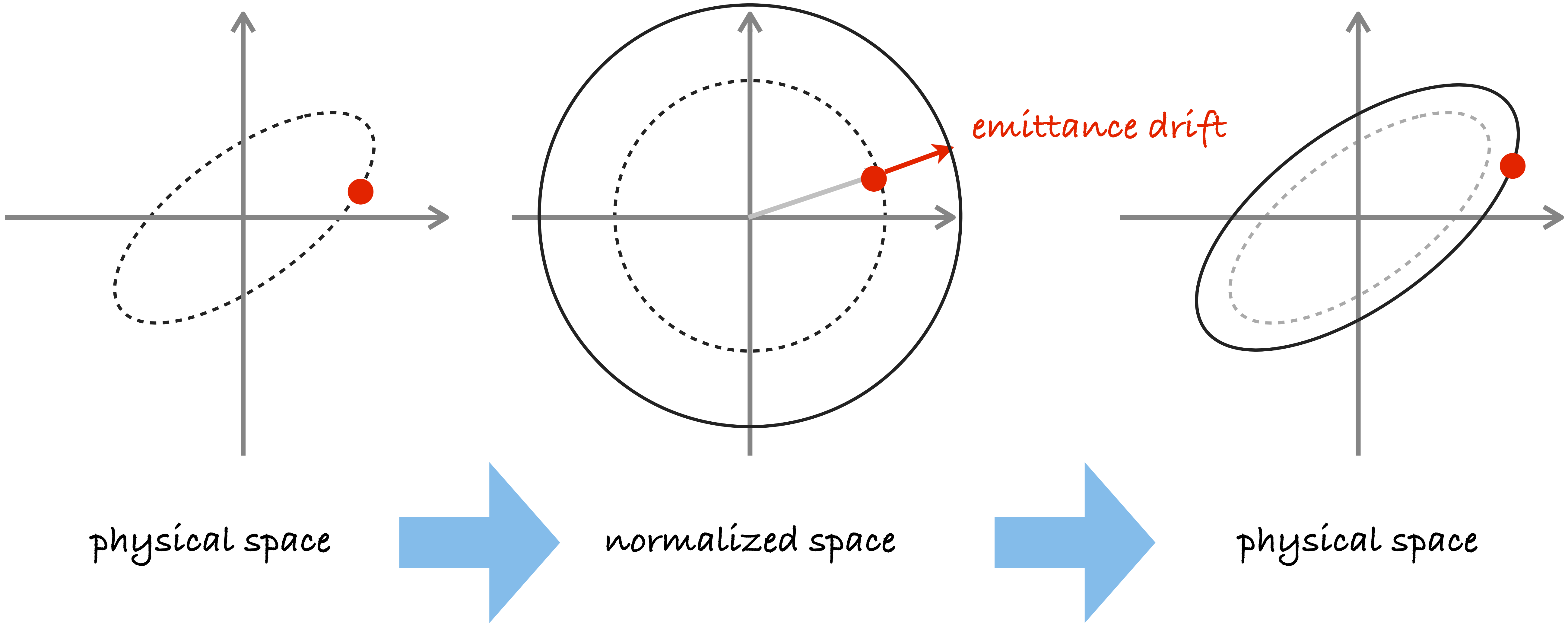
- activated / deactivated through the "write_tm_out" FLAG read in the fort.3

# diffusion kicks in Sixtrack

- (line 7422) inserted in the loop over the elements in the machine sequence and in a loop over particles
- an old emittance drift routine was already there
- the emittance drift contained a bug when transforming to physics to normalized coordinates and back
- the bug has been fixed and the routine expanded

# diffusion kicks in Sixtrack

## old version



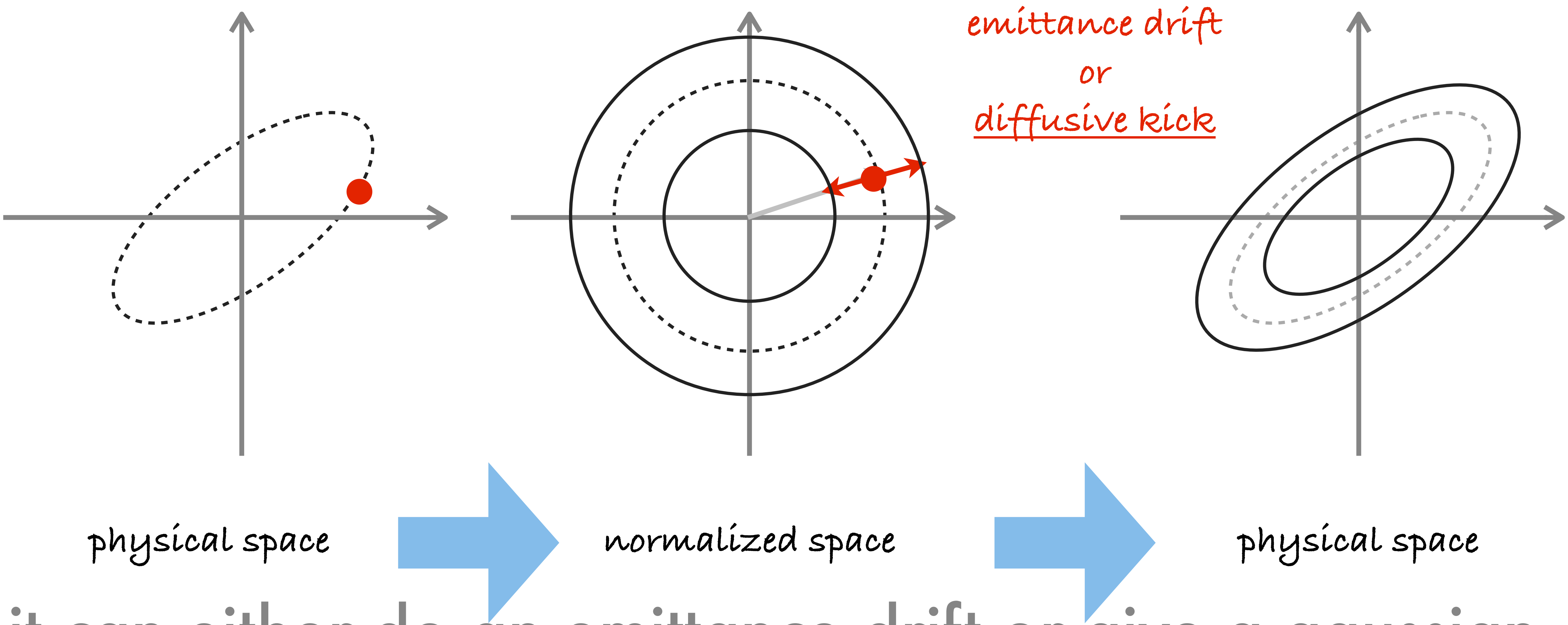
physical space

normalized space

physical space

# diffusion kicks in Sixtrack

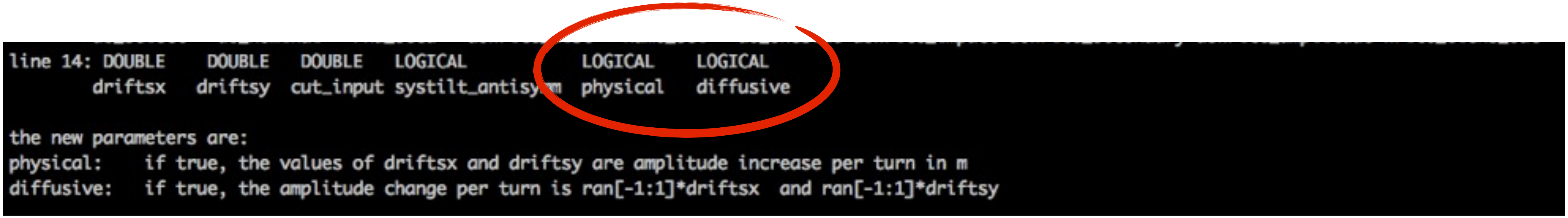
## new version



it can either do an emittance drift or give a gaussian random kick (diffusive behaviour)

# diffusion kicks in Sixtrack

- changes in fort.3 to accommodate the changes in the code:



```
line 14: DOUBLE     DOUBLE    DOUBLE    LOGICAL            LOGICAL     LOGICAL
         driftsx    driftsy   cut_input systilt_antisym   physical    diffusive

the new parameters are:
physical:    if true, the values of driftsx and driftsy are amplitude increase per turn in m
diffusive:   if true, the amplitude change per turn is ran[-1:1]*driftsx  and ran[-1:1]*driftsy
```

- two new logical variables in the line 14 of fort.3
  - 1- it is possible to specify if the “driftsx” and “driftsy” variables are in physical units  [m rad] (TRUE) or in sigma units (FALSE)
  - 2- it is possible to define is the emittance is drifted (FALSE) or if it has a diffusive behavior (TRUE)

**THE ROUTINE HAS NEVER BEEN USED**

# bugs

# (fixed)

- files are opened and closed in “random” places-> fixed (apart from track2.dat, all files are opened now before the loop over the samples)
- distn.dat , efficiency.dat and orbitchecking.dat have the same unit number  (99) and some of them are opened within a sample -> fixed
- file survival.dat was re-written for each particle sample -> fixed
- variable initialization should be done in a consistent way (now it is spread all over the code) -> partially fixed

# suggested improvements

- something which is in "if first run" big cycle, could be done before cycling over the particle samples (more readable)
  EXAMPLE: **read the collimator database** before the loop on particle samples
- associate to each collimator some new flags
  - ICOLL -> ICOLL(MAXN) an array initialized to -1. It would be one flag linking each element to the appropriate collimator in DB (if any) . it could be something like

    ```
    (IN THE CYCLE, in first run)
    do 290 i=1,iu
             if (do_coll && name(i)=coll-type name)
                     cycle over db entries
                     if name(i)==name_db
                     icollmax(i)=database_numebr
            endif
    290 continue
    ```

  - for the aperture

    ```
    NSIG->NSIG(i)
    (ASSIGN APERTURE)
     if(.not. do_nsig) then
            nsig(i) = db_nsig(icollmax(i))
     else
      ... assign accordin to the NAME and the fort.3 settings (ugly but at least only once)
      endif
    ```

# still open questions...

- ANY elements whose name begins with "TC" is treated as a collimator. Any smart way to overcome this?
- it would be better to identify collimator type and family from the fort.2 input (but not retro-compatible)

end (thanks!)

# track.f -> thin6D

| | | | | | |
|---|---|---|---|---|---|
| 4004 | begin of thin6D | | | | |
| 4530 | again more than 500 lines of variable declaration | | | | |
| 4530 | many initialization repeated | | | | |
| 4545 | `firstcoll=.true.` flag for the first collimator | | | | |
| 4549 | `napx=napx00` reset the number in the package | | | | |
| 4561 | if (firstrun) then ===BEGINNING OF FIRST RUN condition, meaning it is the first sample of particles | | | | |
| 4567 | | **READ COLLIMATOR DATABASE** | | | |
| 4576->4673 | | re initialize random generator using the “offset seed” variable and generate random tilts/offsets | | | |
| 4684,4685 | | file opening (twisslike, sigmasettings) | | | |
| 4687 | | loop over elements (do j=1,iu) | | | |
| 4705 | | | IF the beginning of the name is of the coll family | | |
| | | | | associate the collimator opening according to their family | |
| | | | | loop over collimator in the DB | |
| | | | | | if the collimator is in the database (again!) && its length >0 |
| | | | | | apply random gap errors |
| | | | | | calculate 4 normalized gaps (LU,RU, LD, RD) |
| | | | | | associate the appropriate beta function |
| | | | | | write the twisslike file |
| | | | | | apply the offset specified in the DB |
| 4910 | | | | END do | |
| 4915 | | | END if | | |
| 4916 | | END do | | | |
| 4916->4935 | | write more outs | | | |
| 4938->4942 | | re-initialize random generator with random seed | | | |
| 4949->4972 | | re-initialize various flags to zero (again!!) | | | |
| 4976 | | END IF | | | |
| 4980->4996 | re-initalize AGAIN all the flags | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ... | | | | | | | |
| 5001 | **do 660 n=1, numl CYCLE OVER TURNS** | | | | | | | |
| 5008 | | loop over elements (do j=1,iu) (from now on ie=i=element number) | | | | | | |
| 5041->5063 | | | remove particles with high amplitude/angles | | | | | |
| 5067->5072 | | | if first sample, save coordinates of the first particle in variable xbob,xpbob,... | | | | | |
| 5076->5083 | | | Sixtrack stuff (?) | | | | | |
| 5087->5112 | | | **if the name is collimator-type, then set the variable myktrack=1 (not from DB)** | | | | | |
| 5122 | | | if myktrack=1 , go to flag 10 | | | | | |
| 5123->5128 | | | 10 | treatment of the drift space | | | | |
| | | | | if do_coll && the name is collimator-type (AGAIN) | | | | |
| 5146->5260 | | | | | associate the collimator opening according to their family (AGAIN) | | | |
| | | | | | check on first run again | | | |
| | | | | | | misterious check on rselect too... | | |
| | | | | | | cycle over particles | | |
| | | | | | | | transform form general 6track coordinates to collimation-routine | |
| | | | | | | | only for the first particle at the first turn | |
| | | | | | | | initialize some arrays (ALREADY DONE @ LINE 1553!!) | |
| | | | | | | | track the particle down to its coordinates after half collimator length | |
| 5296 | | | | | | | if the particle has not been absorbed jet | |
| | | | | | | | | calculate its amplitude and sum it to amplitude sum |
| 5319 | | | | | | | endif | |
| 5322 | | | | | | end do | | |
| 5326 | | | | | end if | | | |
| 5331 | | | | | check if the collimator is in the database & length>0: then FOUND=TRUE | | | |
| | | | | | assign the variable icoll associated to the element | | | |
| | | | | | ... | | | |

```
5351          if the collimator is in the database
                  assign the variable icoll associated to the element
                  if(.not. do_nsig) assign the DB aperture (checks ONLY NOW)
                  assign the beta function in the DB, if it is the case (if do_nominal)
                  calculate variables for beta beating
5395 -> 5421      if do_write_dist && the collimator is the selected in fort.3
                  -> write the coll ellipse
5426->5457        if firstturn & first sample, write all kind of output
5465              if the collimator is NOT RHIC-TYPE colllimator..
5468                  assign rms errors to aperture nsig
5470                  calculate x max, y max... with two possible beta
5483->5488            assign the DB info to the coll variables
5490->5570            calculate collimator aperture & pencil beam position at coll
5574->5598            if pencil beam && collimator is the pencil beam one && turn=1
                      change collimator tilt (to be parallel with pencil beam)
5604              elseif RHIC
                      special RHIC tratment (not detailed here)
5635              end if
5641->5692        if firstrun && first turn, further outs
5696->5707        c_aperture = 2d0*calc_aperture (double the aperture)
5709->5715        if firstrun & firstturn, write distsec out on collimator number 7??
5719              cycle over particles ( do j = 1, napx)
5720->5733            Copy particle data to 1-dim array and go back to meters
                      set to zero the s position
5738->5744            For zero length element track back half collimator length
5746                  assign flukaname (ipart(j)+100*samplenumber)
5748              end do
5756->5752        if onesided=true then flag the TCP as one sided
```

```
            if the collimator is in the DB (AGAIN!!)
                if the collimator is for RHIC
                    call collimate rhic
                else
                    set TDCQ and TCXRP to one sided
5838                if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
                        ugly series of do cycles (9X)-could be easily merged
5997->6085              for each slice, call COLLIMATE2
6086                else
6088                    call COLLIMATE2
6094                end if (slices)
6097            end if collimator for RHIC
6099        end if (collimator in DB)
6122->6126  initialize impact variables
            cycle over number of particles (do j = 1, napx)
                IF particle hit : part_hit(j).eq.(10000*ie+iturn))
                    For zero length element track back half collimator length
                    copy data back to original verctor (S IS NOT TOUCHED)
6160->6168          Energy update
                else
                    copy back the initial coordinates (necessary??)
6178            end if particle hit
                --- commented code---
6251            IF particle hit : part_hit(j).eq.(10000*ie+iturn))
                    write impacts, if flag is on (do_write_impacts)
                    if particle is absorbed & wirte impacts, another file
6266                if particle is absorbed, write tracks2.dat
                    ...
```

```
                  if the particle has not been absorbed
                    calculate kick
                    assign to adeguate halo family
                  end if particle not absorbed
6319              if  dowritetracks
                    if particel not absorbed
6326                  if particle in some halo & coordinates < 99 and
                      normalized positions are < some cuts
                        ri-transform coordinates in m,rad
                        write tracks.dat
                        write tracks.dat
6369                  end if cuts
6387                end if not absorbed
6389              end if particle not absorbed
                  fill histo variables (number of impacts, average...)
6409              if particle absorbed
                    increase absorbed number
6420                adjust some flags
6421              end if particle absorbed
6425            end if particle hit
6431          END cycle over number of particles
6435->6455     Calculate statistical observables and save into files
6469          IF THE COLL is the selected one
                reset counters for selected collimator
6475->6497      cycle over particles and update selected collimator counters
6502->6529      Calculate average impact parameter and save distribution into file
6548          END IF selected COLL
6645        end if (collimator in DB)
```

```
                else (if do_coll is false, or not collimator name)
                    drift treatment... (not detailed here)
                end if  do_coll && the name is collimator-type (AGAIN)
6752        end if if myktrack=1 (name collimator)
6753    END do loop over elements (do j=1,iu)
9397 end DO cycle over turns
     ... other elements...
```

```
if the collimator is for RHIC
    call collimate rhic
else if ELENS
    call elens routine
else if TM element
    call TM routine
else
    set TDCQ and TCXRP to one sided
    if slices >1 &collimator=TCP,TCSG,TCT,TCLA,TCL
        ugly series of do cycles (9X)-could be easily merged
        for each slice, call COLLIMATE2
    else
        call COLLIMATE2
    end if (slices)
end if collimator for RHIC
```